UNIVERSITY OF TURIN

COMPUTER SCIENCE DEPARTMENT
PHD THESIS

Exploring Transformers: Journey Through Language Processing Architectures and Tasks

Author:
Matteo DELSANTO

Supervisor:
Daniele Paolo RADICIONI

October 21, 2024

UNIVERSITY OF TURIN

Abstract

Computer Science Department

PhD in Computer Science

Exploring Transformers: Journey Through Language Processing Architectures and Tasks

by Matteo DELSANTO

This dissertation illustrates the research activities I carried out during my PhD years. It has as common thread the use of language models and Transformer architectures, that have emerged as the state-of-the-art technology in the Natural Language Processing field. An overview on recent language models is initially provided to frame my research in the context of the modern approaches to the automatic analysis and generation of natural language. Their evolution is traced, by starting from the basic Transformer architecture to advancements such as BERT and GPT-2, up to multimodal models such as OpenAI's GPT-4 and Google's Gemini, and their features are illustrated and discussed. It is then described how the encoder and decoder modules have been exploited for different tasks. Information Extraction from clinical reports and argument mining for detecting grammatical errors are addressed first, as examples of applications relying on transformers encoders. An example of linguistic analysis and categorization is then introduced, targeted at discriminating cognitively impaired subjects from healthy elderly controls: in this case, the analysis is conducted by exploiting a decoder block, whose output is also compared to standard n-gram based language models. Finally, it is shown how to employ the whole transformers architecture to cope with a foundational NLP task, such as word sense disambiguation. The obtained results are discussed and interpreted in the light of the main technological and cultural trends in the NLP field.

Acknowledgements

First of all, I would like to express my deepest gratitude to my supervisor, Daniele Radicioni, for his constant presence, continuous support, and for enriching my research experience, allowing me to grow both as a researcher and as a person.
Thanks to Davide, colleague and friend, for sharing the joys and sorrows of research, and for the unwavering support, even when taking different professional paths.
And finally, thanks to Dario, for being my advisor, my friend, my anchor, *my person*, the one I admire and respect the most in life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 2015 a giant of modern NLP, Christopher Manning, wrote that Deep Learning had produced a tsunami effect on the NLP field (Manning, 2015): in that year Deep Learning technologies had gained a central role in most prominent NLP conferences. At the same time, he cited further influential researchers who expected that soon the NLP field would have been able to measure itself against tasks such as language understanding and generation.

To date, many of those challenges have evolved (Yu et al., 2023); tasks have been re-defined and the so-called downstream tasks have emerged as a testbed to compare different models and approaches (Wang et al., 2018; Wang et al., 2019); either partial or more general solutions have been carried out, and further issues have emerged (Haber and Poesio, 2023). However, as of mid 2024, the evolution of deep architectures can be perhaps best interpreted as an enabling factor with respect to the true revolution in the field. Many and manyfold innovations have been proposed in the last decade, which contributed to attain remarkable results in tasks such as text classification, sentiment analysis, machine translation, question answering, and others; but what contributed most is the introduction of the Transformers architecture, based on attention: the fact that the paper 'Attention is all you need' (Vaswani et al., 2017) represented a point of discontinuity is also evidenced by the 119k citations obtained by April 2024. To date, the feature selection of former Machine Learning systems has been superseded by the *attention* device, that is directly plugged into the system at the architectural level. Also, different from previous sequence models like recurrent neural networks (RNNs) and convolutional neural networks (CNNs), the self-attention mechanisms are highly parallelizable and efficient for capturing long-range dependencies in text data. Such factors, amongst others, determined the possibility to learn rich, contextualized representations of text data. These, in turn, require re-thinking about meaning and sense representations, the status of lexical resources, and how to employ them.

From a historical perspective, in brief it became apparent that the main modules of Transformers (Encoder and Decoder) could be successfully employed for highly different purposes, thanks to different training objectives: while the former modules were trained to predict masked words from context, and fine-tuned for specific downstream tasks such as text classification (Devlin et al., 2018), the latter modules, often referred to as autoregressive models, were trained to predict upcoming words instead of masked words, and mostly employed to generate text in a dialogue setting (OpenAI, 2022).

These architectures became widely adopted in the years around 2020, when I started my doctoral program: this temporal coincidence has thus been a valuable stimulus in pointing my activities towards the exploration of such technologies. My research work targeted applications exploiting either the encoder module, or the decoder module, or both of them. The plan of the dissertation is as follows. An overview on language models is provided in Chapter 2, including the most recent developments, tracing their evolution by starting from the earliest architectures, progressing to the more recent Transformers architecture, and beyond. In the subsequent chapters it is then introduced a series of studies in which transformers are applied to different tasks and problems, assessing how different facets of these models and the various architectural components can be exploited to deal with different applications. In Chapter 3 Transformers are used in more classical fashion: the encoding capabilities of these models are herein considered, and employed to perform tasks such as information extraction, error detection, and related tasks that critically rely on the numerical representation generated starting from text. The focus of this Chapter is on investigating how suited the obtained embeddings are for the mentioned tasks. In Chapter 4 Transformers are *trasformed* into a fully different instrument: the goal is, in fact, to understand how their generative capabilities, adapting to a certain type of language, can be used to analyze language itself. In this case we have preliminarily investigated whether the perplexity, a measure originally conceived for the evaluation of language models, is reliable enough to be employed to analyze language. After experimenting on this issue, perplexity has been used to discriminate people suffering from Alzheimer Dementia from healthy elderly. This was done by acquiring two models based on the linguistic production of these subject classes, and using them to assess the differences that characterize healthy from cognitively impaired language. The chief research question of that study is to explore the reliability of the perplexity as a device to perform linguistic analysis, and to look for the lightest models that ensure reasonable accuracy to assist clinicians in the cognitive assessment of elderly subjects. One final line of research is introduced in Chapter 4: it is concerned with how to use transformers for a long-standing task in Lexical Semantics, Word Sense Disambiguation (WSD). In this case the full architecture is exploited, for building the embeddings and for natural language generation purposes; a novel lexical resource, SE-MACAROON, was created to cope with the WSD task. The underlying research question is in this case to explore whether and how adding a semantic layer on top of a language model can be beneficial to improve its performance on foundational tasks such as WSD.

Chapter 2

Preliminaries

This work focuses on language models as tools to solve a variety of different tasks, exploiting specificities of the different architectures. In this Chapter we will provide background to such topics, following the evolution of language modeling task. Due to the diverse applications of language models explored in this work, each section will be preceded by a dedicated "Preliminaries" or "Background" subsection outlining the specific task at hand.

2.1 Language Modeling task

Formally, the Language Modeling task is defined as the assignment of probability to any possible sequence of words $W_{1,n} = \{w_1 \dots w_n\}$, so to compute $P(W_{1,n})$ (Goldberg, 2017), and Language Models (LMs) are statistical inference tools that allow estimating the probability of a word sequence $W = \{w_1, \dots, w_k\}$ (Manning and Schutze, 1999; Goldberg, 2017). Such probability can be computed through the chain rule of probability,

$$p(W) = \prod_{i=1}^{k} p(w_i | w_1, \dots, w_{i-1}), \tag{2.1}$$

which is customarily approximated as

$$p(W) \approx \prod_{i=1}^{k} p(w_i | w_{i-N+1}, w_{i-N+2}, \dots, w_{i-1}). \tag{2.2}$$

The probabilities assigned by language models are the result of a learning process, in which the model is exposed to a particular kind of textual data. The goal of the learning process is to train the model to predict word sequences that closely resemble the sentences seen during training.

2.2 N-grams

Equation 2.2 can be approximated to take into consideration only the most recent part of the story: for example, the whole story preceding the n-th element of the sequence can be conditioned based only on the previous element,

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-1}).$$

We can generalize from the bigram (which looks one word into the past) to the trigram (which looks two words into the past), and thus to the n-gram (which looks $n-1$ words into the past). The general approximation for n-grams is $P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-N+1:n-1})$, and the probability of a full word sequence can be computed as

$$P(w_{1:n}) \approx \prod_{k=1}^{n} P(w_k|w_{k-1}). \tag{2.3}$$

In this setting, only blocks of few (exactly N) words are considered to predict the whole W: we can thus predict the word sequence based on N-grams, that are blocks of two, three or four preceding elements (bi-grams, tri-grams, four-grams, respectively). In general N-gram models tend to obtain better performance as N increases, with the drawback of making harder the estimation of $P(w_N|W_{1,N-1})$. Another issue featuring these models stems from the fact that when increasing the context size, finding sequences with the same length in the training corpus becomes less likely. In order to deal with N-grams not occurring in the training corpus, called out-of-vocabulary N-grams, language models have to add an additional step of regularization to allow a non-zero probability to be associated to previously unseen N-grams (Gale and Church, 1994; Kneser and Ney, 1995).

2.3 Neural networks: RNNs and LSTMs

Modern neural networks organize neurons into layers, each unit being connected to each unit of the subsequent layer through *synapses* or *edges*. Each layer of the network accepts as an input the output of the preceding layer, performs some transformation of the received data, and produces an output according to the layer architecture. Different layers apply different transformations; edges, in turn, are usually provided with a weight, a real-valued number expressing the strength of the connection among two neurons, which is usually exploited to alter data coming from the preceding layer.

Since neural networks deal with real valued representations of data, we have to extract features from data, which is text in our case, and map them onto a numerical vector representation —usually called embedding— able to correctly grasp the main characteristics of the input data. The role of vector representations is central to neural models, and in fact, modern neural networks are provided with an embedding layer, which is responsible for the creation of a fixed-length vector for each element of the input sequence. It is worth noting that these vector representations mitigate the data sparsity problem by building a continuous space, each word having its corresponding vector in the network space.

Given the relevance of word order in natural language sentences, neural models for NLP have to account for sequential properties in the input sequence. Recurrent Neural Networks (RNNs) are particularly suited to process sequential data such as natural language texts (Elman, 1990). A graphical illustration of RNN model is presented in Figure 2.1: the last hidden state depends on the entire input sequence, that is, the prediction of the next word is conditioned on the previous words in the sentence. The ability of conditioning the prediction of the next word to the preceding context, that is, dealing with sequences, is the most appealing feature of the RNN architectures. Nevertheless, these models struggle to model the context when facing long range dependencies. Unfortunately, however, although they have been conceived to model sequential information, RNNs are not able to model broad range dependencies (Bengio, Simard, and Frasconi, 1994).

Given the difficulties in seizing long range dependencies, RNNs were replaced

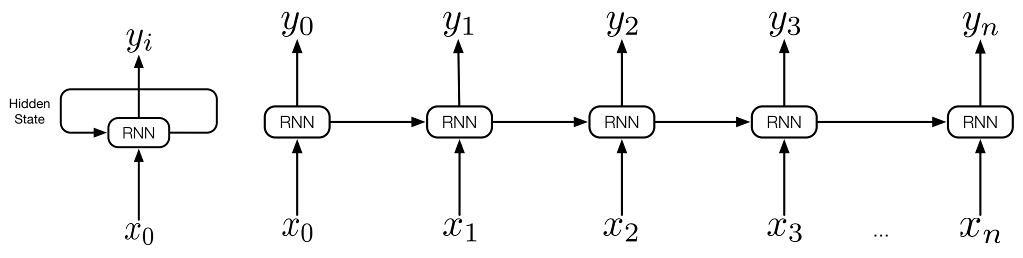


FIGURE 2.1: (Left) RNN unit: takes x_i as input, and computes the output representation y_i as a composition of x_i and the hidden state of the preceding time step. (Right) Representation of an unrolled RNN.

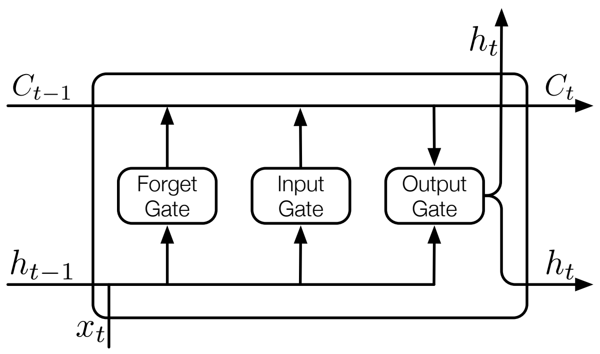


FIGURE 2.2: Representation of an LSTM unit. Here, C_{t-1} and h_{t-1} are the context representation and the hidden state coming from the preceding unit respectively. The input token is represented by x_t. The output of the cell corresponds to its hidden state at the current time step h_t. The updated representation of the context C_t and the hidden state h_t are then forwarded to the next LSTM unit.

by the Long Short-Term Memory networks (LSTM) (Hochreiter and Schmidhuber, 1997). LSTMs are RNNs specifically designed to learn long range dependencies. This is obtained by providing units with an explicit context memory that conveys the information about the preceding context through the time steps. The context representation is achieved through two main operations: (i) forgetting information no longer needed from the context, and (ii) adding new information probably needed for next word prediction. Both sub-tasks are addressed through specialized neural units called gates, which manage the flow of information through the memory state and the output of the LSTM cell. A graphical illustration of an LSTM unit is depicted in Figure 2.2.

2.4 Sequence to sequence

The described structure makes LSTMs particularly suited to deal with sequences and long range dependencies. However, simple LSTM models cannot naturally handle

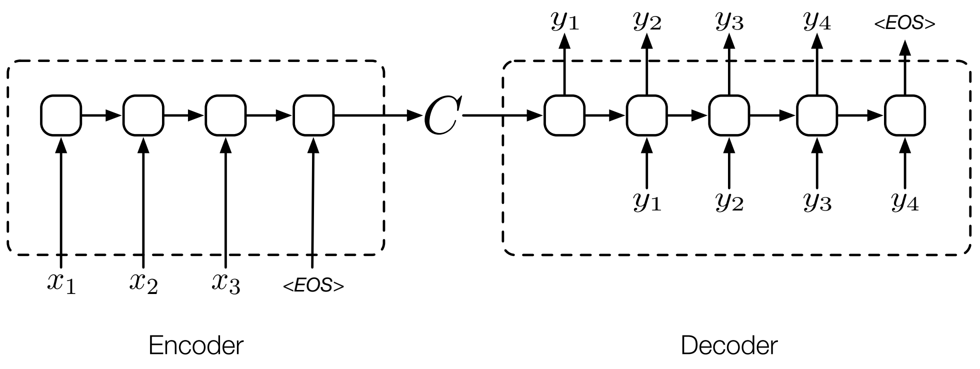


FIGURE 2.3: Representation of a S2S setting. Here <EOS> represents the end of the sentence. The input sequence x_1, x_2, x_3 is processed by the encoder and compressed to the context vector representation C. The context vector is then forwarded to the decoder which predicts the output sequence y_1, y_2, y_3, y_4 by taking as input the previously predicted token at each time step.

tasks in which input and output lengths are not equal, such as machine translation or speech recognition involve dealing with sequences whose length is not fixed beforehand. The Sequence-to-Sequence (S2S) model has been proposed to overcome such limitations (Sutskever, Vinyals, and Le, 2014). The S2S model relies on LSTMs to map an arbitrary length sequence $x_1, \ldots, x_n$ to another sequence $y_1, \ldots, y_k$ where k may be different from n. In this setting, the input sequence is processed by an encoder, which compresses the sequence to a fixed length vector representation C. The decoder is then initialized on the C vector, and predicts the output token by token, accounting for the previously predicted token at each time step. A graphical representation of the S2S architecture is depicted in Figure 2.3.

2.5 Transformers architecture

Despite the ability of LSTM architectures to deal with long range dependencies, these models still struggle in representing larger pieces of text and suffer from high training time due to the recurrent connections which build these units. Additionally, the S2S architecture suffers from the loss of informative load in compressing the whole input sequence into a single fixed length vector representation. Transformers (Vaswani et al., 2017), together with the attention mechanism (Bahdanau, Cho, and Bengio, 2014), alleviate these problems by both increasing the amount of exploited information from the context, and getting rid of the recurrent connections.

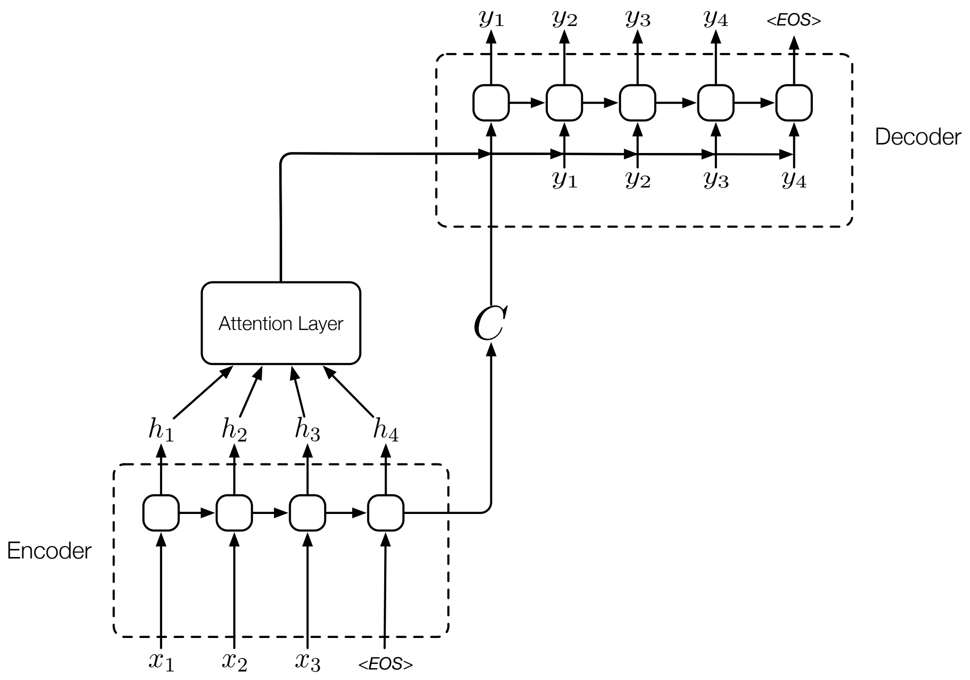


FIGURE 2.4: Attention mechanism in a S2S setting. Here each prediction of the decoder relies on both the previously predicted token and a composition of the encoder hidden states.

The attention mechanism has been designed to alleviate the difficulties in S2S models; this is done by allowing the decoder to directly exploit the encoder's hidden states rather than just using the final context representation provided by the encoder itself. Adopting an attention mechanism allows the model to selectively focus on parts of the input that are likely to be the most useful for the task at hand. The attention mechanism is particularly suited to address tasks which need to take decisions relying on specific parts of the input data. An illustration of the attention mechanism fitted in the S2S setting is depicted in Figure 2.4. Attention mechanisms plays a key role in the Transformer architecture; in particular this model follows the S2S design pattern where the encoder processes the input sequence, the output is then forwarded to the decoder which is concerned with the output predictions. In this Section we will refer to the encoder-decoder model as to the Transformer block (Figure 2.5). Since Transformers get rid of recurrent connections, thereby allowing models to deal with sequences, the encoder represents the input through a combination of word embeddings and information about the position of words in the input sentence: in so doing, the model is able to account for ordering information. After this first operation, the encoder unit is made of an attention layer followed by a

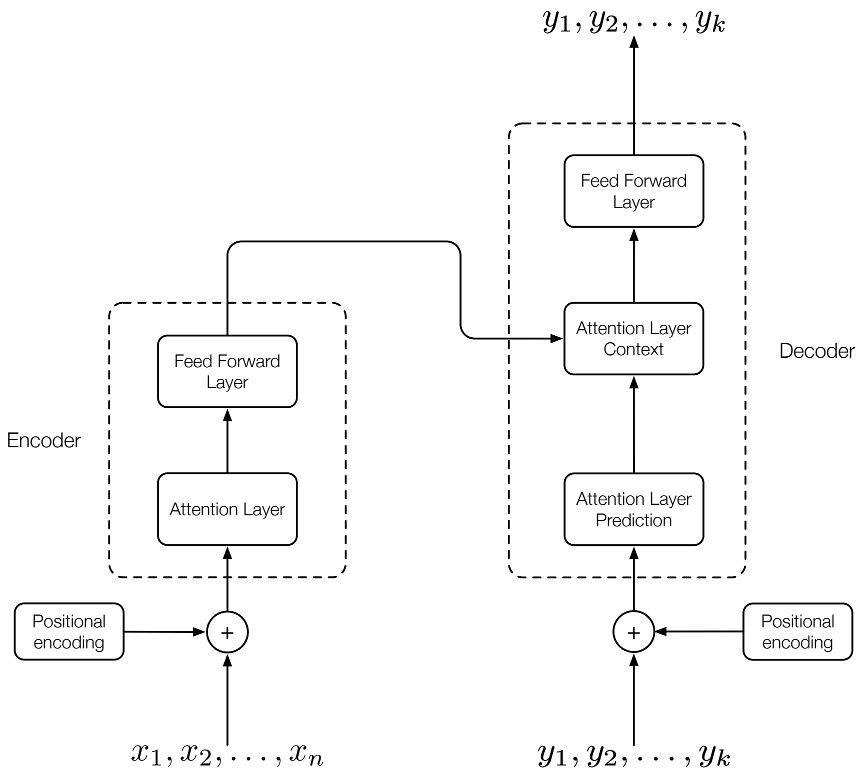


FIGURE 2.5: High level representation of transformer block. The input sequence $x_1, x_2, \ldots x_n$ is combined with positional information to account for ordering properties. The input is then processed from the attention layer of the Encoder and a simple neural layer aimed at representing the whole input sentence. The Encoder output is then combined with previously predicted tokens from the Decoder through another attention layer, and then, the last layer computes the output representation for each input token.

simple neural layer which is responsible for computing the context representation. The decoder unit combines the previously predicted word representations with the positional information to keep track of the order of the words, and sends forward these vectors through an attention layer that is aimed at selecting the most useful information among the predictions. After these first steps the decoder combines the information from previously predicted tokens with the context representation, coming from the encoder, through another attention layer. Lastly, a simple neural layer is concerned with computing the output representation. Most popular models consist of several Transformer blocks stacked one on another: this allows the model to increase its abstraction capabilities (as the number of stacked blocks grows, the representation that can be calculated is more and more abstract (Tenney, Das, and Pavlick, 2019; Tenney et al., 2019)).

Transformers have been widely adopted and improved to address diverse Natural Language Understanding benchmarks, such as those in the GLUE (Wang et al.,

2018) and SuperGLUE (Wang et al., 2019) benchmarks. The successful adoption of models such as BERT (Devlin et al., 2018) and GPT-2 (Radford et al., 2019) in different application settings has attracted considerable efforts on improving such models (Lan et al., 2019; Liu et al., 2019; Yang et al., 2019; Raffel et al., 2019; Brown et al., 2020).

This two models also represent two different ways of exploiting Transformers architecture: the first focused on the encoding part, while the second on the decoding part.

2.5.1 BERT and BERT-like architectures

BERT is a large language model based on Transformers: its key innovation is applying bidirectional training to Transformers, and consequently, to language modeling. Using BERT involves to deal with two different rather important phases: *pre-training*, where the model is exposed to unlabeled textual data so as to learn the main features from the type of employed texts; and *fine-tuning*, where the pre-trained model is fine-tuned to address a specific task by exploiting labeled data. Additionally, since BERT's aim is to produce contextual representations of the input language, only the encoder mechanism from Transformers is necessary. During the pre-training phase BERT is designed to address two unsupervised predictive tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP).

The MLM is what gives BERT the bidirectional attribute: in this setting, a small percentage of the input tokens —15% in the released models— is masked and the training objective is to predict those masked tokens. For example, the sentence *I shot an elephant in my pajamas* may be rewritten into *I shot an [MASK] in my pajamas* and the BERT target is to predict the word *elephant* istead of the *[MASK]* token. As we can read in Devlin et al., 2018, MLM objective allows representing both left and right context, thereby making the training bidirectional, but at the cost of misaligning the pre-training with the fine-tuning: the *[MASK]* token does not appear during fine-tuning. To mitigate the misalignment issue, the pre-training process does not always replace the masked word with the *[MASK]* token. For that 15% of input token to be masked, the replacement occurs as follows: 80% of the time, the token is replaced with the *[MASK]* token, 10% of the time, it is replaced with a random token from the vocabulary, and 10% of the time, it remains unchanged. The loss function is than computed by accounting for the masked tokens only, thus ignoring the prediction on non-masked words.

The NSP training objective has been devised to allow BERT to learn the relationship between sentences. The NSP involves selecting pairs of sentences A and B: in half cases the sentence B directly follows the sentence A, while in the remaining half the sentence B is randomly selected within a textual corpus. The purpose of the training objective is to learn whether sentence B directly follows A or not. Since many NLP tasks involve learning the relationship between sentences, such as, for example, Recognizing Textual Entailment (RTE) or Question Answering (QA), the NSP objective is to strengthen the model by precisely learning the relationship between pairs of sentences.

The BERT architecture follows the Transformer's design pattern reported in Figure 2.5: the difference is that BERT is made of stacked Transformers encoder blocks, one on another, to increase the abstraction level. Stacking encoder blocks not only allows dealing with more and more abstract representations, but also to reproduce a complete NLP pipeline with different BERT layers deal with different linguistic levels (Tenney, Das, and Pavlick, 2019; Tenney et al., 2019). In Figure 2.6 we report a graphical illustration of the BERT model's input as provided by the authors (Devlin et al., 2018). Given that BERT has to deal with pair of sentences as well as masked tokens, the authors, provided the model with two special tokens, the *[CLS]* token is placed at the beginning of the first sentence and is used from following classification layers placed on top of BERT, while the token *[SEP]* is used as separator, to mark the end of a sentence. The sentence embeddings reported in the figure are used to distinguish between the sentence A and B for the NSP task, while the positional information is accounted through the positional embeddings.

Once the pre-training of the model has been completed, the model may be fine-tuned to address a specific NLP task. For each downstream NLP task a fine-tuning process is needed, in particular, to specialize a pre-trained model is sufficient to plug a classification layer on top of BERT relying on the encoded representation for words as well as for the [CLS] token. For example, a sentence pair classification task such as QA or RTE, a simple classification layer may be plugged on top of the Transformer's

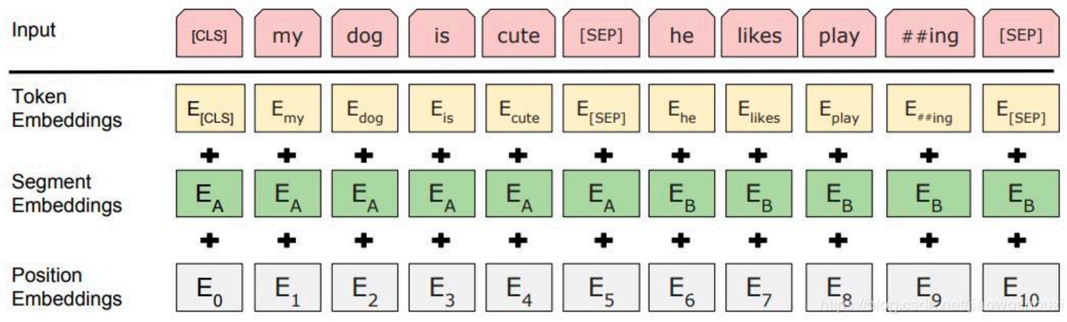

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

FIGURE 2.6: Representation of the BERT model's input. Token embeddings represent the word vectors, positional embeddings represent encode the ordering information of words in sentences. The segment embeddings are used to distinguish between the two input sentences. The [SEP] special token is used to mark the end of a sentence, as separator, while the token [CLS] can be used for classification purposes. Figure borrowed from (Devlin et al., 2018).

output relying on the [CLS] token. We refer the readers to Devlin et al. (2018) for an exhaustive list of design pattern for different kind of task.

BERT was just the first in a series of "BERT-like" models, which adjusted some of BERT's hyperparameters or training modalities (such as training pipeline or data, or both) to achieve improved performance on specific tasks or to add new capabilities to the model, like multilingual support. ROBERTA (Liu et al., 2019) and XLM-ROBERTA (Conneau et al., 2019) are just two examples of this trend.

2.5.2 GPT2

GPT-2 is a language model based on Transformers and trained to predict the next word given the preceding context (Radford et al., 2019). GPT-2, like traditional language models, predicts one token at a time, and the new prediction is appended to the input sequence for next time step. Inspired by the work in (Liu et al., 2018), where the Transformer-Decoder architecture was proposed, GPT-2 is made of stacked decoder units only. More precisely, while the Transformer-Decoder block is very similar to the decoder of the Transformer architecture, it gets rid of the encoder unit and of the contextual attention layer in the decoder. A graphical illustration of the GPT-2 architecture is depicted in Figure 2.7.

The GPT-2 model has been trained on 40GB of Internet text carefully selected for quality, that is a selection of documents curated or supervised by humans. One main trait featuring the training data selection is that many different domains have been exploited as data sources; this allows the neural network to model language

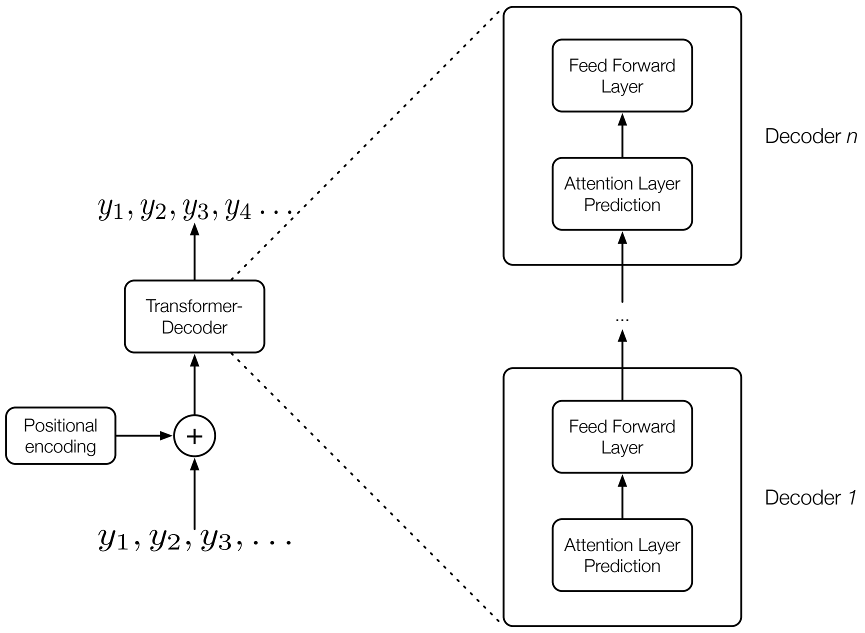


FIGURE 2.7: Representation of GPT-2 architecture. The model is made of n stacked decoder blocks. The input sequence $y_1, y_2, y_3, \ldots$ is processed by the n stacked encoders that form the Transformer-Decoder block. The last decoder produces the next token y_4 that is appended to the input sequence for the next time step. Each decoder is made of an attention layer dedicated to processing the input sequence and a simple neural layer that computes the output representation.

properties avoiding a strong polarization towards a specific domain. Additionally, it is worth noting that the number of stacked decoder units impacts performance, in that increasing the number of levels produces an improvement on the language modeling capabilities.

2.6 Modern Large Language Models

Neural Language Models continued to evolve faster and faster over the last few years, becoming "Large" Language Models, advanced language models with massive parameter sizes and exceptional learning capabilities, like GPT-3 (Brown et al., 2020), PALM (Chowdhery et al., 2023) and LLAMA-2 (Touvron et al., 2023). The core module behind these models is always the Transformers architecture, which, as confirmed by Chang et al. (2023), have revolutionized the field of NLP with their ability to handle sequential data efficiently, allowing for parallelization and capturing long-range dependencies in text, particularly thanks to the Attention Module. But key elements characterizing modern Large Language Models are the in-context learning ability (Brown et al., 2020), and the Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Ziegler et al., 2019). The first, enables LLMs to generate more coherent and contextually relevant responses, making them suitable for interactive and conversational applications, while the second involves fine-tuning the model using human-generated responses as rewards, allowing the model to learn from its mistakes and improve its performance over time (Chang et al., 2023). Such capabilities, acquired by models thanks to these new type of training, also gave birth to novel approaches to interact with LLMs, like prompt engineering, where users design and provide specific prompt texts to guide LLMs in generating desired responses or completing specific tasks (Clavié et al., 2023; White et al., 2023; Zhou et al., 2022).

2.6.1 LLama2

Among these models, LLama2 is an Open Source model, evolved from a former LLama model (Touvron et al., 2023); it is based on a collection of pretrained and fine-tuned large language models (LLMs) ranging in scale from 7 to 70 billion parameters; it employs the standard Transformers architecture with some optimizations to improve performance. As a LLM, LLama2 makes use of RLHF, which also requires a data collection step: the typical interaction involves asking annotators to first write a prompt, and then choosing between two sampled model responses, based on some given criteria. In order to maximize the diversity, the two responses to a given prompt are sampled from two different model variants, by also varying the *temperature* hyper-parameter. In addition to giving participants a forced choice, it also asks annotators to label the degree to which they prefer their chosen response over the alternative: their choice is either *significantly better*, *better*, *slightly better*, or *negligibly better/ unsure*. The underlying reward model takes a model response and its corresponding prompt (including contexts from previous turns) as inputs and produces as output a scalar score to indicate the quality (e.g., helpfulness and safety) of the model generation. To train the reward model, the collected pairwise human preference data is converted into a binary ranking label format (i.e., chosen & rejected); the chosen response is associated to a higher score than its counterpart, using a binary ranking loss consistent with:

$$L_{\text{ranking}} = -\log(\sigma(r_\theta(x, y_c) - r_\theta(x, y_r) - m(r))), \tag{2.4}$$

where $r_\theta(x, y)$ is the scalar score output for prompt x and completion y with model weights θ, y_c is the preferred response that annotators choose and y_r is the rejected counterpart (Ouyang et al., 2022). Finally, margin $m(r)$, is a a discrete function of the preference rating: given that the preference ratings are decomposed as a scale of four points (e.g., *significantly better*), it can be useful to leverage this information to explicitly teach the reward model to assign more discrepant scores to the generations that have more differences. σ represent the sigmoid function. The LLama2 training corpus includes a new mix of data from publicly available sources; additionally, the authors removed data from sites that are known to contain a high volume of personal information about private individuals, and trained on 2 trillion tokens of data as this approach provides a good performance–cost trade-off, up-sampling the most factual sources in an effort to increase knowledge and dampen hallucinations (Touvron et al., 2023).

2.6.2 Multimodality

Finally, it is important to mention a further evolution of language models, based not only on text, but on a series of inputs acquired in different "modalities": multimodal language models. Multimodal deep learning has been successfully proposed in different studies, like in (Ngiam et al., 2011), where a technique for using deep autoencoders to learn features from audio and video was developed. Kiros, Salakhutdinov, and Zemel (2014) went more in the direction of Multimodal Language Models, proposing to jointly learn word representations and image features by training models together with a convolutional network. The most recent and popular developments in this field are GPT-4 (OpenAI, 2023) and Gemini (Anil et al., 2023), models capable of processing images and text, in the first case, and also audio and video clips in the second one, producing text as output (also images in the case of Gemini).

Gemini models build on top of Transformer decoders that are enhanced with improvements in architecture and model optimization to enable stable training at scale and optimized inference on Google's Tensor Processing Units (Anil et al., 2023). They are trained to support 32k context length, employing efficient attention mechanisms. Gemini models (Ultra, Pro and Nano, based on the model size) are trained to accommodate textual input interleaved with a wide variety of audio and visual inputs, such as natural images, charts, screenshots, PDFs, and videos, and they can produce text and image outputs. These models are trained on a dataset that is both multimodal and multilingual, that uses data from web documents, books, and code, and includes image, audio, and video data. A graphical illustration of the Gemini architecture is depicted in Figure 2.8.

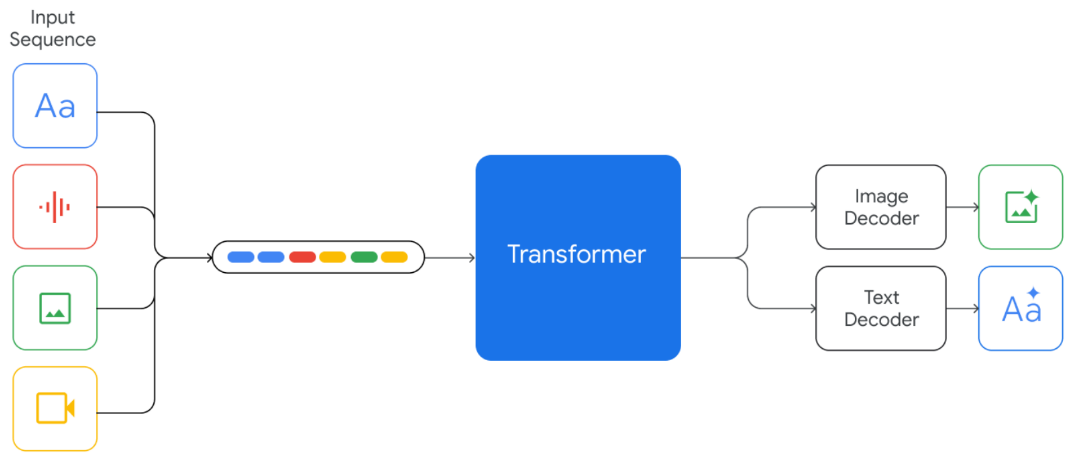


FIGURE 2.8: Representation of Gemini architecture from the Anil et al. (2023) technical report. The model supports interleaved sequences of text, image, audio, and video as inputs (illustrated by tokens of different colors in the input sequence). It can output responses with interleaved image and text.

Chapter 3

Language models as Encoders

This chapter delves into the encoding capabilities of language models: as a matter of fact, information extraction, error detection, and related tasks critically rely on the numerical representation, embeddings, that these models can generate from text. In the following sections we are going to explore the results obtained using language models on two different types of texts, and employing different extraction techniques: in (Mensa et al., 2022) we worked on clinical reports asking the model to produce a single or multiple label for the given input, while in (Colla, Delsanto, and Di Nuovo, 2023) we framed the problem of Grammatical Error Detection as a token classification task, employing a sequence labelling strategy based on the BIO labelling schema, to provide a label for each token within the input sequence.

3.1 Road Accidents: Information Extraction from Clinical Reports

In recent years Electronic Health Records (EHRs) have become more and more common, leading to the collection of increasing amounts of health and clinical data. Among these, Emergency Room records are particularly interesting, since they can provide structured data concerning the patient, combined with unstructured free-text data describing the events that led to the hospitalization. These records are usually categorized under the type of event that caused the patient injuries, allowing for multiple analysis centered on the specific events. The focus of this work is in particular on road traffic accidents. A road traffic accident (RTA) is defined by the French National Institute of Statistics and Economic Studies as *an accident that occurs* *when at least one road vehicle is involved in an accident which happens on an open public road, and at least one person ends up being killed or injured* (Économiques, n.d.).

This sort of data is naturally suited for the analysis of RTAs, although often lacking of information on the specific injuries reported by the patients.

In this section we investigate a novel approach for the study of RTAs: we extract relevant information concerning road accidents from medical records, so to develop a system providing detailed statistical insights on the injuries possibly caused by a given accident; to the best of our knowledge this task was never previously addressed in literature. The feasibility of such a system requires however to determine if current state-of-the-art language models (e.g., BERT) are at least able to identify and extract which vehicles are involved in a medical report. More specifically, we are interested in the extraction of the vehicles involved in the accident, and in which vehicle the patient was in when the accident occurred.

This precious information may then enrich patients' data, typically including age, gender, and the reported injuries. The whole extraction process needs to be completely automated, since asking the medical operators to promptly collect such data (maybe by choosing a vehicle in a complex list of tens of options) is completely unfeasible, due to the conditions of high time pressure and workload customarily afflicting emergency departments.

3.1.1 Related Work

Many works have been published on the topic of RTAs, showing that the problem can be analyzed from different points of view and with different aims. A great deal of effort has been invested in accident severity prediction (Chong, Abraham, and Paprzycki, 2005; Santos et al., 2021; Assi, 2020; AlMamlook et al., 2019; Li et al., 2012) and traffic accident anticipation (Bao, Yu, and Kong, 2020), aimed at the understanding of which factors such as weather conditions, time, road quality, etc. contribute the most to the severity of the accidents. Other efforts have been spent in comparing different methods to understand which ones are better suited for the modeling of RTAs. In particular, a recent study compares many machine learning approaches including naïve Bayes, logistic regression, K-nearest neighbors, AdaBoost, support vector machines, and random forests finding the latter one as the best suited for the task (Bokaba, Doorsamy, and Paul, 2022).

Particular emphasis has also been put on the explainability of the adopted algorithms: in (Parra, Ponce, and Rodrigo, 2020) random forests and decision trees achieved good results, providing experimental evidence for the fact that weather conditions are related to car accidents. In Das et al., 2021 a substantial sub-category of accidents has been examined, specifically addressing those against poles and trees. The authors rely on text mining and on other interpretable machine learning techniques to analyze available crash narratives, so to complement the results with explanations. In the literature on RTAs, works typically examine a specific dataset which is released by a particular State or Region of the world. A plethora of such datasets can be found, for instance the National Road Network dataset from Canada (Government of Canada, Accessed: 10-10-2022), the Road Safety Data dataset from the UK Department of Transport (UK Department for Transport, Accessed: 10-10-2022), the Montreal Vehicle Collisions dataset from the City of Montreal (City of Montreal, Accessed: 10-10-2022), the Setùbal (Portugal) reports (Santos et al., 2021), the Gauteng (South Africa) reports from the Gauteng Department of Community Safety (Bokaba, Doorsamy, and Paul, 2022), etc. The amount of information provided by these datasets is highly variable. The crash reports from Victoria, Australia (Assi, 2020) also include the gender and age of the drivers. Among the various surveyed datasets, the one from Louisiana treated in (Das et al., 2021) is a rare instance of a dataset listing some insights concerning the gravity of the injuries incurred by the drivers, which are classified as *fatal*, *incapacitating*, *non-incapacitating*, *possible*, and *no injury*. A complete review of many of these datasets can be found in (Gutierrez-Osorio and Pedraza, 2020).

Our work is hardly comparable with the current state-of-the-art, since we do not rely on categorical and numerical data in our analysis, but rather focus on text extraction from free-text narratives included in clinical data. The application of NLP techniques in this field is however not completely new, since in recent years some works have been carried out trying to extract and classify RTAs from social media (Salas, Georgakis, and Petalas, 2017; Salas et al., 2017). In summary, the novelty of our approach stems from the radically different dataset that we are employing, which requires specific NLP techniques to be dealt with. With this work we are allowing for the future extraction of detailed information regarding injuries in accidents, which can be paired with the data computed through more traditional approaches aimed at a better understanding of the impact of RTAs.

3.1.2 Dataset and Annotation

The data employed in the present study are real-world Emergency Room Reports (ERRs) collected in Italian Hospitals, and then made available by the Italian National Institute of Health in the frame of the SINIACA project (Pitidis et al., 2014). The SINIACA project[1] is the Italian branch of the European Injury Database (EU-IDB), an EU-wide surveillance system concerned with accidents, collecting data from hospital emergency department patients according to EU recommendation Lyons, Kisse, and Rogmans, 2015. The SINIACA-IDB is a data collection on injuries, based on a sample of hospital emergency departments, in implementation of the recommendation of the Council of the European Union no. C 164/2007/01 on injury prevention and safety promotion.

The original dataset consists of 153, 826 clinical records from the SINIACA-IDB, which were originally annotated by hospital staff as referring to road traffic accidents or not. In this work we only consider the 35, 952 records that concern RTAs. It is important to note that these reports are very challenging to process. ERRs are compiled by medical staff under huge time pressure, which leads to typos and disjointed fragments of text, at times resembling bullet lists rather than actual sentences. By randomly sampling and analyzing 592 records of the dataset we measured that over 10% tokens contain either typos, abbreviations, or acronyms (on average 2.25 per record) (Mensa et al., 2020a; Mensa et al., 2020b). All of these elements significantly increase the difficulty of extracting relevant data from the records.

Data annotation. In this preliminary work we focus on the extraction of two types of information. Namely, given a record, we want to retrieve which kind of vehicles were actually involved in the incident (*Task 1*), and in which vehicle the patient was

[1]'Sistema Informativo Nazionale sugli Incidenti in Ambiente di Civile Abitazione', National Information System on Accidents in Civil Housing Environment.

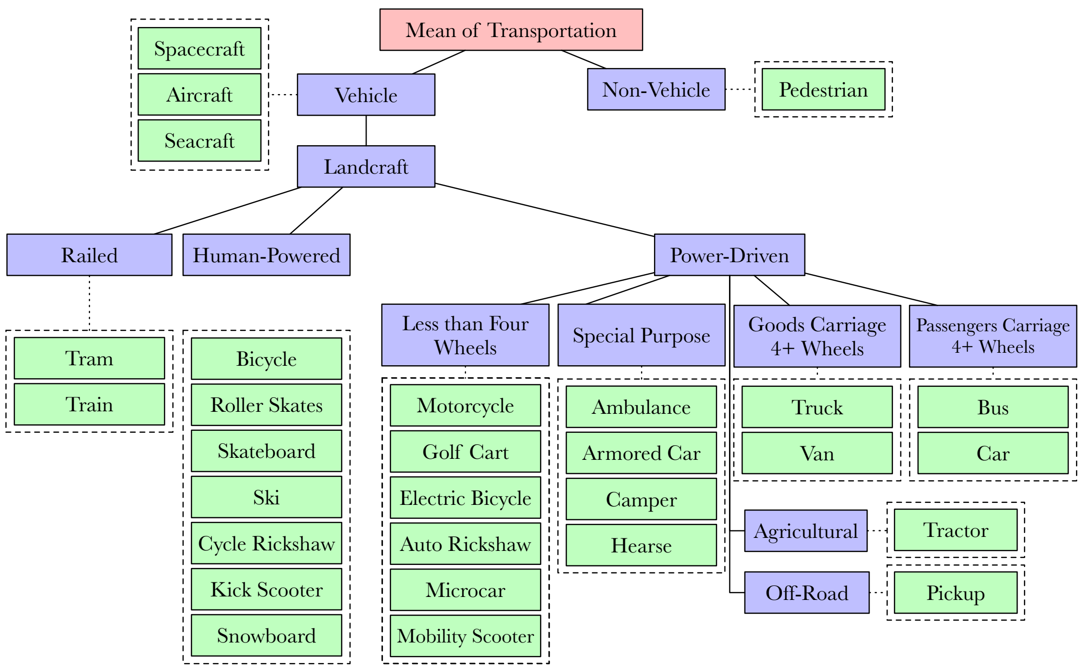


FIGURE 3.1: Taxonomy of vehicles adopted for the annotation process.

when the accident occurred (*Task 2*). The dataset was annotated separately for these two tasks, that will be referred to as T1 and T2, respectively. The set of labels used to annotate vehicles for both tasks has been designed based on the Wikipedia page related to vehicle categories,[2] which in turn relies on the UNECE (United Nations Economic Commission for Europe) categories and regulations (UNECE, n.d.). Some classes were merged for simplicity, while the PEDESTRIAN class was added to account for the annotation of pedestrians. The resulting taxonomy is illustrated in Figure 3.1.

Only the leafs of the taxonomy were adopted as labels throughout the annotation process; however, just a few were actually found in our dataset. Table 3.1 (left side) reports the label distribution on the two tasks for the 1,000 randomly selected records. Provided that testing for the classification of labels with very few occurrences (e.g., one or zero for T2) is not meaningful, we decided to remove from the test set the 13 entries labeled as VAN/TRUCK/TRACTOR/AUTO-RICKSHAW/ARMORED-CAR for either T1 or T2. Table 3.1 (right side) illustrates the final dataset of 987 records. Future work will focus on the annotation of entries containing less common vehicles, so to be able to evaluate the system for the classification of this type of labels. As illustrated in the Table, in the T2 task only one label per entry was annotated (since the patient could only be in one vehicle), while T1 allowed for multiple labels since

[2] https://en.wikipedia.org/wiki/Vehicle_category.

TABLE 3.1: Label distribution for the medical records annotated, before (left - 1,000 records) and after (right - 987 records) pruning the under represented labels.

Label	# in T1	# in T2
NS (not specified)	405	412
CAR	392	284
MOTORCYCLE	186	173
PEDESTRIAN	78	77
BUS	30	27
BICYCLE	26	24
VAN	5	1
TRUCK	5	0
TRACTOR	1	1
AUTO-RICKSHAW	1	1
ARMORED-CAR	1	0
Total	1,130	1,000

Label	# in T1	# in T2
NS	405	412
CAR	386	279
MOTORCYCLE	185	172
PEDESTRIAN	74	73
BUS	30	27
BICYCLE	26	24
Total	1106	987

multiple vehicles can be involved in one accident. Moreover, the NS (not specified) tag allows for the classification of records that do not provide enough information to determine which vehicles were involved in the accident. The annotation process was carried out by two annotators on the text annotation tool Doccano (Nakayama et al., 2018). The resulting Inter Annotator Agreement – calculated as Cohen's kappa coefficient– was of 0.9957 for T1 and 0.9930 for T2. Such high IAA shows that this task is quite easy for humans; however, we will show that for artificial systems some language nuances are still difficult to decipher, especially for T2.

3.1.3 System and Evaluation

Our system is based on a multilingual version of the BERT neural model by Devlin et al. (2018).[3] We employed the same model for the resolution of both T1 and T2. More specifically, the tasks are defined as follows: in T1 the system is requested to retrieve all the vehicles involved in the RTA, while in T2 the system has to detect the mean of transportation used by the patient at the time of the accident. We fine-tuned through MLM the existing language model on the whole set of 153,826 records for ten epochs so to specialize the model on the type of language used (Italian, deeply

[3]`https://huggingface.co/bert-base-multilingual-cased`.

blended with medical jargon). Finally, we stacked a linear classifier from an off-the-shelf Transformers library from HuggingFace (Wolf et al., 2020) on top of the pre-trained model, which exploits the dense representation of the input record to classify it with the appropriate label(s). The final model employs two different types of loss function according to the task definition: in the case of T1 we employed the Binary Cross-Entropy loss, that is the model is allowed to classify multiple labels for each entry (more specifically, we have N loss functions, one for each class), while for T2 we adopted the Cross-Entropy loss, since in this setting the model must return only one class. The evaluation was conducted on the 987 annotated entries described in Section 3.1.2, in a 10-fold setup, with a training of 10 epochs for each such fold.

The system has been also been evaluated against a simple baseline based on string-matching for both T1 and T2: for each label l a set of trigger vehicles and their synonyms S^l has been obtained from the Treccani Italian Dictionary (Treccani, n.d.). For T1 the baseline classifies a record as l if the record contains a trigger word from the corresponding set S^l. For T2 the baseline works similarly, but the trigger word has also to be in the proximity of a *trigger expression*. Trigger expressions are a collection of words and phrases (e.g., *was driving*, *guiding*, *run over*, *passenger of*, etc.) often used in the records to indicate the fact that the patient was either the driver or a passenger in a vehicle. The baseline has been run on the same 10 folds as the BERT system, discarding the training portion of each fold, since no training was performed for this system.

Results and Discussion. Results for the T1 and T2 tasks are reported in Table 3.2 and Table 3.3 respectively. Concerning T1, both BERT and the baseline perform well overall. However, depending on the class, the two systems show very diverse performances. The CAR and PEDESTRIAN classes appear to be more challenging compared to the MOTORCYCLE class: this is due to the linguistic variability adopted by the medical personnel when describing car or pedestrian accidents. As an example, PEDESTRIAN accidents are mostly described as 'hit by vehicle', however, some specific instances such as 'grazed by a rearview mirror', 'slided against a car', 'crushed by a tire', etc. can also be found. Such variability seems to be well dealt with by the neural model, while the baseline falls short in managing this complexity. On

TABLE 3.2: Precision (P), Recall (R) and F1 score on the T1 task (10 fold).

Label	Baseline			BERT		
	P	R	F1	P	R	F1
NS	0.64	1.00	0.78	0.98	0.96	**0.97**
CAR	0.99	0.44	0.60	0.95	0.96	**0.95**
MOTORCYCLE	1.00	0.82	0.90	0.96	0.98	**0.97**
PEDESTRIAN	0.90	0.26	0.38	0.89	0.73	**0.80**
BUS	1.00	0.93	**0.96**	1.00	0.70	0.82
BICYCLE	0.90	0.77	**0.82**	1.00	0.50	0.67
All (weighted)	0.86	0.72	0.72	0.96	0.93	**0.94**

the other side, BUS and BICYCLE accidents are so few in the dataset that the neural model cannot properly generalize, whilst the baseline can deal with them by simply matching the words *bicycle* and *bus*, which are included in the respective trigger sets. The overall performance on the dataset (bottom row) shows that BERT mostly succeeds in this task with high accuracy (0.94 F1 score). BERT is also very precise on all classes (even on NS), which is a key feature for our goal: being able to rule out the records where no relevant information is available (thus limiting false negatives) is a priority.

The results on T2 show how much the language model is able to distinguish vehicles involved in the incident from those in which the patient was either a passenger or the driver. PEDESTRIAN is the most challenging class: by looking at the records, this is probably due to the fact that the notion of being a pedestrian is sometimes expressed in a convoluted and less clear way, compared to the notion of being passenger or driver. Increasing the dimension of the training set may be helpful to solve this issue. Given the huge drop in the performances of the baseline, this task is evidently too complex for a string-matching approach. Once again, the overall performance on the dataset (bottom row) shows that BERT can successfully deal with the task (0.92 weighted average F1 score). By looking specifically at precision and recall, we observe that the system is more precise in finding T1 occurrences, but it lacks recall. This phenomenon is interestingly reversed in T2, which may be explained by the fact that the notion of driving or being a passenger is more complex to detect.

TABLE 3.3: Precision (P), Recall (R) and F1 score on the T2 task (10 fold).

Label	Baseline			BERT		
	P	R	F1	P	R	F1
NS	0.48	1.00	0.65	0.96	0.93	**0.95**
CAR	0.86	0.13	0.22	0.92	0.93	**0.92**
MOTORCYCLE	1.00	0.23	0.36	0.96	0.97	**0.97**
PEDESTRIAN	0.80	0.65	0.71	0.75	0.73	**0.74**
BUS	0.11	0.04	0.06	0.79	0.85	**0.82**
BICYCLE	0.20	0.04	0.07	0.76	0.92	**0.83**
All (weighted)	0.68	0.54	0.45	0.92	0.92	**0.92**

3.1.4 Road Traffic Accidents Insights

In this Section we report some insights on the whole dataset, obtained by running the BERT system on the 35,952 entries concerning RTAs. In order to improve the stability of the system, we used all of the 987 entries as training set. We presently discuss the results on the T2 task (concerned with the vehicle on which the patient was traveling at the time of the accident), which it is definitely more interesting. To these ends, we provide aggregated data on gender and age, deferring the investigation on injuries to future work. Figure 3.2a reports the distribution of road accidents for each age interval with respect to the vehicles involved. According to such figures, the age of people involved in car and motorcycle accidents spans from 21 to 60. Differently, bus accidents mainly involve people over 60 years old, while 21% of bicycle accidents concern people from 41 to 50 years old. In contrast, accidents involving pedestrians seem to mainly concern lower and higher bounds of the age range: almost 50% of such collisions concern people under 30, and 25% involves over 70. In Figure 3.2b we illustrate the distribution of road accidents for each vehicle class with respect to the age interval of the patient. People aged between 0 and 10 are involved in accidents mainly as car passengers (53%) and pedestrians (29%), probably due to the fact that children are typically accompanied by parents. Conversely, people aged from 11 to 20 are often involved in motorcycle accidents (42%), consistently with the changes in travel habits during adolescence. The car becomes the main vehicle involved in accidents up to the 50 years limit, while only about the 30% of accidents concerns motorcycles. Progressively, after the age of 50, the

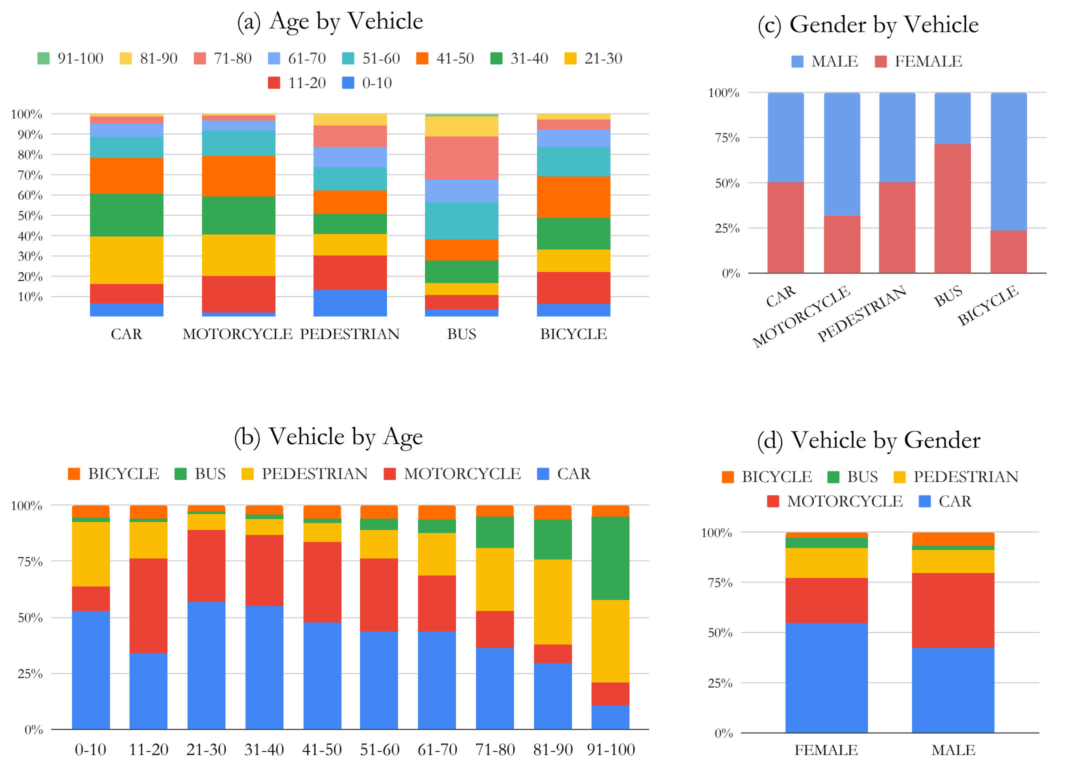


FIGURE 3.2: Statistical insights on road traffic accidents obtained by the full dataset. The reported statistics refer to the T2 task i.e. the vehicle in which the patient was in at the time of the accident.

percentage of car accidents decreases consistently with the aging of the patients; at the same time accidents involving pedestrians and buses increase accordingly to the changes in movement habits. Figure 3.2c shows the distribution of road accidents by gender with respect to the vehicles involved. Accidents involving cars and pedestrians are equally distributed between males and females, while events where the patient travels by bus mainly concern females (71%). Motorcycle (68%) and bicycle (77%) accidents mostly involve males. Figure 3.2d reports the distribution of road accidents for each vehicle class with respect to the gender of the accident patient. The 55% of accidents involving females concern cars, while the 22% involve motorcycles. Consistently with such figures, the events involving males often concern cars (42%), followed by motorcycle accidents (38%). Accidents involving pedestrians are of the same order of magnitude for both genders, while males experience three times more bicycle accidents than females.

3.2 ELICODE at MultiGED2023: fine-tuning XLM-RoBERTa for multilingual grammatical error detection

Grammatical Error Detection (GED) is the task of automatically identifying errors in learner language. Despite its name, the errors to be identified are not only grammatical errors, but different error types are considered, e.g. spelling, punctuation, lexical. In Second Language Acquisition and Learner Corpus Research, indeed, an error is defined as "a linguistic form or combination of forms which, in the same context and under similar conditions of production, would, in all likelihood, not be produced by the speakers' native speaker counterparts" (Lennon, 1991). As can be noticed, this definition includes different causes, i.e. grammaticality and correctness, or acceptability, strangeness and infelicity (James, 1998). This difference results in different resources annotating different errors, with some annotating as grammatical errors also appropriateness errors—i.e. pragmatics, register and stylistic choices (Lüdeling and Hirschmann, 2015, p. 140)—others excluding appropriateness, but including orthographical and semantic well-formedness together with acceptability (Di Nuovo, 2022).

In both GED task and the related Grammatical Error Correction (GEC) task, research has focused mainly on learner English (as second or foreign language) (Bell, Yannakoudakis, and Rei, 2019; Ng et al., 2014; Bryant et al., 2019). Recently, also non-English error-annotated data sets have been released (Boyd, 2018; Náplava et al., 2022). Thanks to these recent trends, the authors of MultiGED (Volodina et al., 2023) organised in 2023 the first multilingual GED shared task, hosted at the workshop series on Natural Language Processing for Computer-Assisted Language Learning (NLP4CALL).

Both GED and GEC can be seen as low or mid-resource tasks, because of three main characteristics: requiring time-expensive and highly-specialised human annotation, annotated data sets are usually small in size; the incorrect tokens in a text are significantly scarce if compared to the correct ones; since errors pertain to different error categories, each error type in the data sets is represented unevenly.

The data sets included in MultiGED shared task are in Czech, English, German, Italian and Swedish. Some of these data sets have been already used for GED or

GEC tasks—i.e. *Falko* and *Merlin* corpora (Boyd, 2018), *Grammar Error Correction Corpus for Czech* (GECCC) (Náplava et al., 2022), *First Certificate in English* (FCE) corpus (Yannakoudakis, Briscoe, and Medlock, 2011)—others have been released *ad hoc* for this shared task—i.e. *Russian Error-Annotated Learner English Corpus* (REALEC) (Kuzmenko and Kutuzov, 2014), released only as development and test data sets, and learner Swedish *SweLL-gold* (Volodina et al., 2019), comprising training, development and test data sets.

The aim of MultiGED is to detect tokens to be corrected labelling them as correct or incorrect, performing a binary classification task at token level. Training and development data sets were segmented into sentences and tokens (no information at text level was released).

Following previous GED shared tasks, the used evaluation metric is F0.5, which weights precision twice as much as recall, carried out on the Codalab competition platform.[4]

The authors of the shared task encouraged submissions using a multilingual approach and additional resources, provided that these resources are publicly available for research purposes. However, since different resources can annotate different errors, the use of other additional data might be a double-edged sword. In fact, the additional data would increase the tool's ability to identify a greater variety of errors, but at the same time, as the tool is evaluated in-domain, it moves away from the characteristics of the test set.

3.2.1 Related work

The detection of errors in interlanguage texts (Selinker, 1972) is a challenging task that has received significant attention in the natural language processing community, since GED systems can be used to provide feedback and guidance to language learners. In this section, we review some of the most relevant and recent studies in this area and in the related task of GEC.

Initially tackled using rule-based approaches, GED systems have evolved from being able to identify only certain types of errors to being more and more able to

[4]https://codalab.lisn.upsaclay.fr/competitions/9784

handle the complexity and variability of natural language, thanks to modern machine learning techniques which make use of large annotated text corpora, usually released in the occasion of shared tasks. This switch is evident in the evolution of the shared task from CoNLL-2013 (Ng et al., 2013) to CoNLL-2014 (Ng et al., 2014), when it changed from annotating only five error types to *all* error types.[5]

In CoNLL-2014 shared task, the majority of the systems made use of hybrid approaches able to deal with all error types together, as compared to previous year's submissions, where a specific classifier per each error type was trained. The most popular approaches made use of one or more of the following: the Language Model (LM) based approach (using n-gram language models), which has been used for both GED and GEC; the phrase-based Statistical Machine Translation (SMT) approach, used mainly for GEC; and rule-based approaches to tackle regular error types.

In 2019, the Building Educational Applications (BEA) shared task on GEC (Bryant et al., 2019) introduces a new data set, joining the Cambridge English Write & Improve, W&I (Yannakoudakis et al., 2018) and LOCNESS corpus (Granger, 1998), making the test data set bigger than the one on which CoNLL-2014 systems were tested (from 50 essays on two different topics, to 350 essays on about 50 topics). Another major change concerns the use of neural machine translation (Bryant et al., 2022)—being it based on recurrent neural networks (Bahdanau, Cho, and Bengio, 2014), convolutional neural networks (Gehring et al., 2016), or transformers (Vaswani et al., 2017)—instead of SMT and n-gram-based LMs. BEA reported results highlighted that the same system had different performances in texts at different CEFR levels (Little, 2006), lexical errors were the most difficult to detect and correct, and multi-token errors were better handled than in the previous shared task.

Bell, Yannakoudakis, and Rei (2019) integrate contextual embeddings—BERT, ELMo and Flair embeddings (Peters et al., 2017; Devlin et al., 2018; Akbik, Blythe, and Vollgraf, 2018)—in Rei (2017) architecture for GED (a bi-LSTM sequence labeler at token and sentence level, making use also of character-level bi-LSTM, to benefit from morphological information). Their best model used BERT embeddings and

[5]Twenty-eight error types are annotated in the CoNLL-2014 benchmark data set. However, it should be noticed that this is still far from annotating all error types. For example, in the English Corpus of Learner English (ICLE) (Granger et al., 2020) there are 54 error tags, in the error-annotated learner Italian corpus, VALICO-UD (Di Nuovo, 2022, p. 94), 120 error tags.

proved to better generalise in out-of-domain texts. Their analyses show that missing tokens are the most difficult errors to indentify.

Kaneko and Komachi (2019) proposed an extension of BERT base (Devlin et al., 2018) with multi-head multi-layer attention, since research has shown that different layers are best-suited for different tasks, e.g. lower layers capture local syntactic relationships, higher layers longer-range relationships (Peters et al., 2018).

Recently, Yuan et al. (2021) fine-tuned BERT, XLNet (Yang et al., 2019) and ELECTRA (Clark et al., 2020) models to perform GED in English. The three models obtained the new state of the art in binary GED training on FCE data set and testing on BEA-dev, FCE-test and CoNLL-2014, with ELECTRA performing the best overall. Thus, they used ELECTRA to carry out some multi-class GED experiments to boost performance on GEC data sets using it as auxiliary input or for re-ranking.

Our system treats GED as a binary sequence labelling task, like all the above-described systems, and since the best results have been obtained by fine-tuning transformer-based models, we followed this approach by fine-tuning XLM-RoBERTa model (Conneau et al., 2019). We decided to use multilingual RoBERTa because its training focuses on the discrimination of the masked token, and thus, it is conceptually similar to GED. In the following section we quantitatively analyse MultiGED data set, before describing in detail our submitted systems in Section 3.2.3.

3.2.2 Data set quantitative analysis

MultiGED data set contains labelled training and development sets in Czech (GECCC), English (FCE), Italian (Merlin), German (Falko and Merlin) and Swedish (SweLL-gold). In particular, for English language an additional data set (REALEC) has been released only as development set. In addition, for each data set an unlabelled test set has been released.

Following the work by Siino et al. (2022), we quantitatively analyse the 5-language data sets using established corpus linguistics methods implemented in Sketch Engine (Kilgarriff et al., 2014).[6] We report general data set figures in Table 3.4, as computed using Sketch Engine.

[6]Available at the URL https://www.sketchengine.eu (last accessed on 28 March 2023).

TABLE 3.4: MultiGED data set in figures. # stands for *number of*.

Source corpus	**Language**	**Split**	**# Tokens**	**# Unique wods**
GECCC	Czech	train	333,995	37,228
		dev	32,071	8,145
		test	35,075	8,764
FCE	English	train	465,038	13,972
		dev	35,463	3,569
		test	42,545	3,800
REALEC	English	train	–	–
		dev	88,698	6,208
		test	90,391	6,300
Falko-MERLIN	German	train	306,847	20,561
		dev	39,627	5,606
		test	36,763	5,478
MERLIN	Italian	train	82,040	6,957
		dev	9,326	2,041
		test	10,300	2,176
SweLL-gold	Swedish	train	115,547	10,791
		dev	15,713	3,225
		test	14,666	3,141

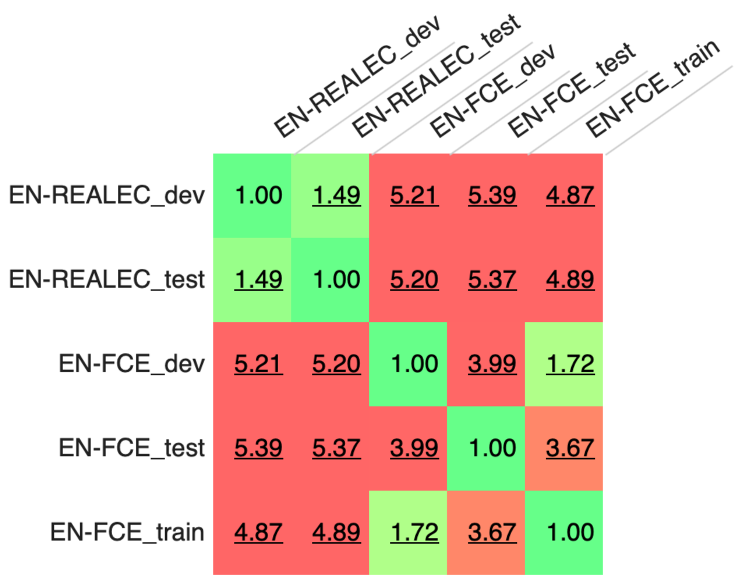
We used Compare Corpora, the built-in function of Sketch Engine that applies chi-square (χ^2) test (Kilgarriff, 2001), to compare training, development and test sets per each language. The result of this comparison is a confusion matrix per each language, reported in Figure 3.3, showing values greater or equal to 1, with 1 indicating identity. The higher the value, the larger the difference between the compared data sets.[7] For English we created a comprehensive confusion matrix comparing the two different corpora (FCE and REALEC).

Looking at the matrices, we could suppose that systems should have less trouble in handling the task in German, Czech, Swedish (in order) than in Italian and English.

English (EN) data set – Since the big difference between FCE and REALEC, the lowest results should be obtained using models trained on FCE and tested on REALEC. Better results could be instead obtained fine-tuning in-domain using REALEC development set and testing it on the test set (because of the smaller similarity score between REALEC development and training sets).

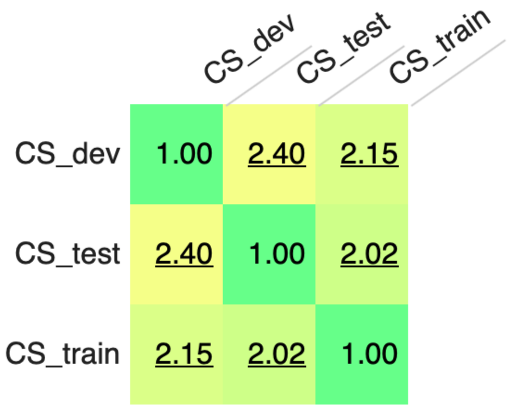
It is interesting to notice that REALEC development and test data sets have a similarity score (i.e. 1.49) significantly lower than FCE development and test data sets

[7]Please consider that correct or incorrect labels are not taken into account in this comparison. This comparison, instead, gives as an idea of how different the data sets are according to the different words used. Compare Corpora tool is affected by set size: this is why development and test sets, being the smallest, have a higher similarity score than when compared individually to the bigger training sets.



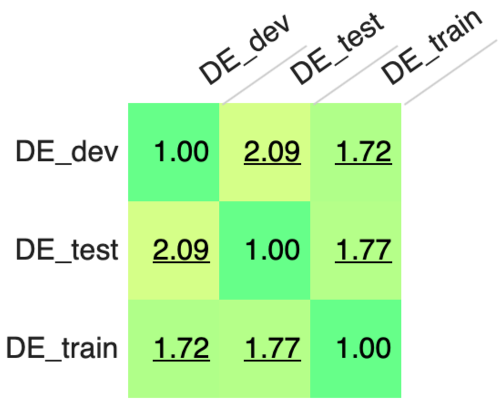
	EN-REALEC_dev	EN-REALEC_test	EN-FCE_dev	EN-FCE_test	EN-FCE_train
EN-REALEC_dev	1.00	1.49	5.21	5.39	4.87
EN-REALEC_test	1.49	1.00	5.20	5.37	4.89
EN-FCE_dev	5.21	5.20	1.00	3.99	1.72
EN-FCE_test	5.39	5.37	3.99	1.00	3.67
EN-FCE_train	4.87	4.89	1.72	3.67	1.00

(A) EN data sets.



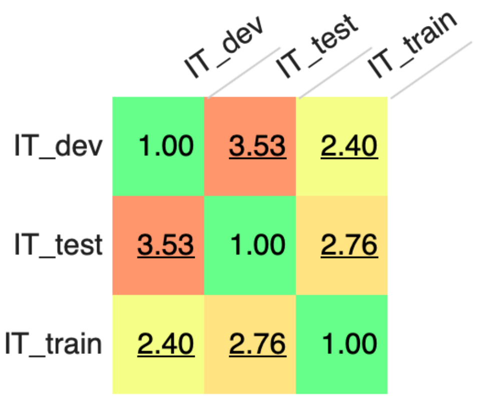
	CS_dev	CS_test	CS_train
CS_dev	1.00	2.40	2.15
CS_test	2.40	1.00	2.02
CS_train	2.15	2.02	1.00

(B) CS data set.



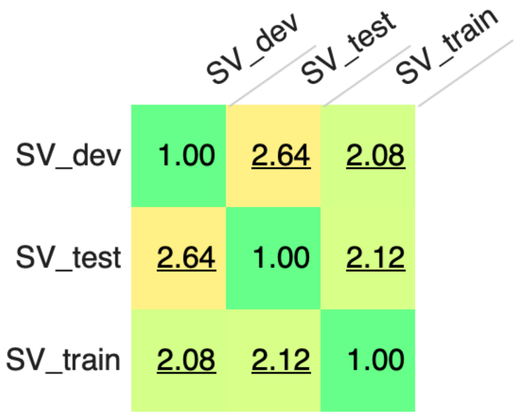
	DE_dev	DE_test	DE_train
DE_dev	1.00	2.09	1.72
DE_test	2.09	1.00	1.77
DE_train	1.72	1.77	1.00

(C) DE data set.



	IT_dev	IT_test	IT_train
IT_dev	1.00	3.53	2.40
IT_test	3.53	1.00	2.76
IT_train	2.40	2.76	1.00

(D) IT data set.



	SV_dev	SV_test	SV_train
SV_dev	1.00	2.64	2.08
SV_test	2.64	1.00	2.12
SV_train	2.08	2.12	1.00

(E) SV data set.

FIGURE 3.3: Confusion matrices obtained with word-based chi-square test. The value 1.00 indicates identity between the compared data sets. The greater the value, the more different the data sets.

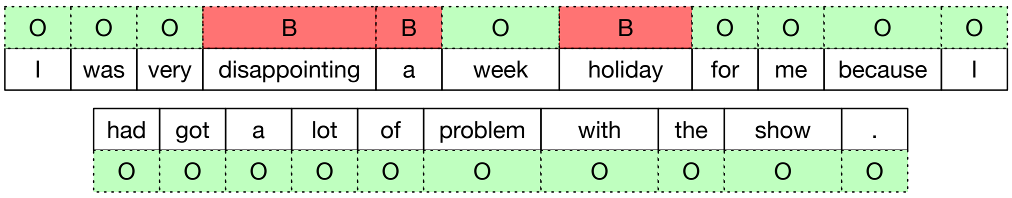

O O O B B O B O O O O
I was very disappointing a week holiday for me because I
had got a lot of problem with the show .
O O O O O O O O O O

FIGURE 3.4: The output of the model for the sentence *I was very disappointing a week holiday for me because I had got a lot of problem with the show.* Here the token *disappointing* is marked as the beginning of an error unit. By the same token, *a* is marked as beginning of a new error due to the information loss caused by the conversion from error-tagged corpora to binary token labelling. The token *holiday* is also marked as an incorrect use. The other tokens are marked as correct uses.

(i.e. 3.99). FCE training and development data sets have a similarity score of 1.72. FCE training and test data sets of 3.67. These results might suggest that the English data set is challenging for the models.

Czech (CS) data set – The lower similarity scores between the data sets suggest that systems should perform better on Czech than in English test set. Also if compared to the similarity scores obtained in Italian data sets, the lower similarity scores might indicate that the systems should perform better on Czech than in the Italian test set.

German (DE) data set – Since the low similarity score, indicating a bigger similarity between the sets, should mean that German should be the easiest to tackle for the models.

Italian (IT) data set – Here again, since similarity scores between the sets are lower than in English one, models should perform better on the Italian data set than in the English. In addition, the higher similarity score between development and test data sets suggests that choosing the best performance model according to the results on the development set should be avoided. Instead training on both training and development data sets should ensure the best performance in this data set.

Swedish (SV) data set – According to the reported similarity scores, Swedish training set is in an order of similarity with development and test sets as the Czech sets. This might suggest that similar performances might be expected.

3.2.3 System description

Given the nature of the MultiGED shared task, we framed the problem as a token classification task, where systems are required to provide a label for each token within the input sequence. More precisely, we employed a sequence labelling strategy using the BIO labelling schema (Ramshaw and Marcus, 1999). The standard schema is formed by B-I-O tags, where each token in a sentence is labelled with one of the three tags: B indicates the beginning of the error span, i.e. the first token of an incorrect use; I is used to label tokens inside the error unit; O marks tokens that are out of the error span, hence correct. However, since in our task we did not have information about the number of errors nor the error span, we decided to use always B to mark an incorrect token, even when preceded by another incorrect token, and O to mark the correct tokens.

The adopted model allows framing the problem as token classification task that, given a sentence $W = w_1 w_2 \ldots w_n$, amounts to labelling each word w_i with B or O tags because of the above-mentioned reason. Figure 3.4 reports an example of the system output of a sentence from the English FCE training data. Considering the example, we can see that the token *disappointing* is correctly tagged with B, indicating an incorrect usage, and then it is followed by another incorrect token *a*—marked again with the label B because of the information loss from the conversion from error-tagged corpora to binary token labelling. In the same example, the token *week* is labelled as correct while the token *holiday* is labelled as incorrect token.

The model we employed is based on XLM-RoBERTa large: we stacked a linear classifier—with input size of 1024 units and the output size is set to the number of labels—on top of the pre-trained XLM-RoBERTa model, inserting in between the two a dropout layer—with a dropout probability set to 0.1—to avoid overfitting. Finally, in order to compute the distance between the actual data and the predictions we adopted the Cross Entropy loss function. The model architecture is depicted in Figure 3.5.

To run the experiments, we devised two different experimental settings. In the first one, we trained the models only on the provided training set for 10 epochs, using the development set to select the model achieving the best performance across

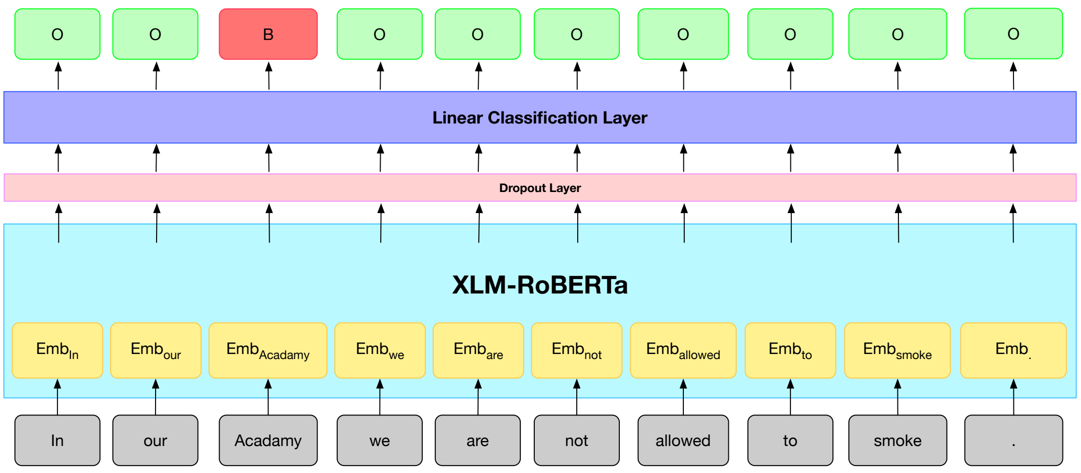

O
O
B
O
O
O
O
O
O
O
Linear Classification Layer
Dropout Layer
XLM-RoBERTa
Emb_{In}
Emb_{our}
$\text{Emb}_{\text{Acadamy}}$
Emb_{we}
Emb_{are}
Emb_{not}
$\text{Emb}_{\text{allowed}}$
Emb_{to}
$\text{Emb}_{\text{smoke}}$
$\text{Emb}_{.}$
In
our
Acadamy
we
are
not
allowed
to
smoke
.

FIGURE 3.5: Graphic representation of the model. The grey boxes represent the tokens in the example. These tokens are vectorised and converted into embeddings by XLM-RoBERTa. Tokenisation in XLM-RoBERTa is simplified in this figure for readability reasons. XML-RoBERTa output is inputted to the linear classifier, after passing a dropout layer. The classifier predicts the label B or O for each token.

the training epochs (ELICODE). In the second setting, we trained each model jointly on the training and development sets for 10 epochs, and retained the 10-epoch trained model (ELICODE$_{ALL}$).[8]

To implement our models, we started from the ClinicalTransformerNER framework by Yang et al., 2020, which responded to the majority of our needs (also if created for a different field), and we adapted the code so to make it work with XLM-RoBERTa language model.[9] In particular, as reported in their Github repository[10], the ClinicalTransformerNER package is the implementation of a transformer based NER system for clinical information extraction task, with the aim to provide a simple and quick tool for researchers to conduct experiments with NER systems. In particular, the great merit of this package is to be like an interface, with the possibility to run experiments independently from the specific transformer implementation (BERT, ROBERTA or others), which can be chosen through command line parameters together with training parameters.

Our experiments were performed on machinery provided by the Competence Centre for Scientific Computing (Aldinucci et al., 2017). In particular, we exploited nodes with 2x Intel Xeon Processor E5-2680 v3 and 128GB memory. The training

[8]In both experimental settings we adopted a batch size of 4 and an early stop of 5 epochs.
[9]Code and models are publicly available at `https://github.com/davidecolla/EliCoDe`
[10]ClinicalTransformerNER Github repository - `https://github.com/uf-hobi-informatics-lab/ClinicalTransformerNER`

time is about 15 hours per epoch for the provided languages with a large training data—i.e. Czech, English and German—and drops to 8 hours per epoch for Italian and Swedish. The time taken in the prediction phase is about 25 minutes per language.

3.2.4 Results and discussion

We report in Table 3.5 the results obtained by all teams participating to MultiGED shared task (upper part of the table),[11] and additional experimental results—i.e. a baseline and our submitted models but trained in a multilingual fashion (bottom part of the table). As far as the baseline is concerned, we extracted the token counts from the training data and adopted the multinomial Naive Bayes classifier for sequence labelling (Baseline). As far as the multilingual models are concerned, we followed the same experimental settings of the submitted monolingual models, training two multilingual models: a first model trained only on the concatenation of training data sets (ELICODE$_{MLT}$), the second concatenating also the development data sets (ELICODE$_{MLT_{ALL}}$).

The overall results obtained by both ELICODE and ELICODE$_{ALL}$ are higher than those obtained by the other competing systems, except for the English REALEC test set.

Concerning Precision (P), the baseline and both our ELICODE and ELICODE$_{ALL}$ submissions perform well overall. However, on the FCE partition of the English data set the scores consistently decrease by about 10% and, as expected, the REALEC partition is the most challenging data set: Precision scores drop from about 80% on average to about 40%. As far as Recall (R) is concerned, the token count-based baseline performs poorly: the average Recall of the baseline across languages is about 12% while the average score of ELICODE and ELICODE$_{ALL}$ is about 58%. Following the same trend as Precision, Recall scores for both our submitted systems drop from about 62% of average to 40% on the REALEC English data set. Given the definition of F0.5 metric—i.e. it puts more importance on Precision with respect to Recall—, the overall scores reflect the trend of Precision: the average F0.5 score is about 76%

[11]We took the results from the official MultiGED repository: `https://github.com/spraakbanken/multiged-2023`.

TABLE 3.5: Results of experiments in the token classification task. To increase readability, we partitioned the results on two tables grouped by language. We reported the results for all the systems submitted to the MultiGED competition—in the upper part of each sub-table—together with the results of our submission (ELICODE and ELICODE$_{ALL}$). The bottom part of each sub-table report the Naive Bayes-based baseline and the multilingual models (ELICODE$_{MLT}$ and ELICODE$_{MLT_{ALL}}$) results. For each system we report the scores obtained on all the languages included in the competition; for each language, the corresponding columns report the Precision (P), Recall (R) and F0.5 scores. The highest F0.5 scores are in bold.

System	Czech			English - FCE			English - REALEC		
	P	R	F0.5	P	R	F0.5	P	R	F0.5
DSL-MIM-HUS	58.31	55.69	57.76	72.36	37.81	61.18	62.81	28.88	**50.86**
Brainstorm Thinkers	62.35	23.44	46.81	70.21	37.55	59.81	48.19	31.22	43.46
VLP-char	34.93	63.95	38.42	20.76	29.53	22.07	-	-	-
NTNU-TRH	80.65	6.49	24.54	81.37	1.84	8.45	51.34	1.13	5.19
su-dali	-	-	-	-	-	-	-	-	-
ELICODE	82.29	50.61	73.14	73.64	50.34	67.40	44.32	40.73	43.55
ELICODE$_{ALL}$	82.01	51.79	73.44	71.67	50.74	66.21	43.69	40.74	43.07
Baseline	85.69	21.19	53.26	72.81	7.55	26.69	36.40	5.67	17.46
ELICODE$_{MLT}$	83.06	50.72	**73.66**	73.85	50.08	67.45	44.36	42.29	43.93
ELICODE$_{MLT_{ALL}}$	82.79	49.56	73.01	75.01	48.94	**67.79**	45.34	40.29	44.23

System	German			Italian			Swedish		
	P	R	F0.5	P	R	F0.5	P	R	F0.5
DSL-MIM-HUS	77.80	51.92	70.75	75.72	38.67	63.55	74.85	44.92	66.05
Brainstorm Thinkers	77.94	47.55	69.11	70.65	36.46	59.49	73.81	39.94	63.11
VLP-char	25.18	44.27	27.56	25.79	44.24	28.14	26.40	55.00	29.46
NTNU-TRH	83.56	15.58	44.61	93.38	19.84	53.62	80.12	5.09	20.31
su-dali	-	-	-	-	-	-	82.41	27.18	58.60
ELICODE	83.87	71.89	81.16	85.63	66.69	81.03	80.56	67.50	77.56
ELICODE$_{ALL}$	84.78	73.75	**82.32**	86.67	67.96	**82.15**	81.80	66.34	78.16
Baseline	80.99	10.25	34.02	85.11	10.72	35.65	78.09	13.65	40.16
ELICODE$_{MLT}$	83.47	72.52	81.02	85.30	69.64	81.63	82.24	65.94	78.36
ELICODE$_{MLT_{ALL}}$	84.80	71.09	81.65	85.71	65.95	80.87	83.34	64.37	**78.70**

for both ELICODE and ELICODE$_{ALL}$ on all languages but the English REALEC data set, where the average F0.5 drops to 43%.

Considering the different languages, as expected from the quantitative analysis from Section 3.2.2, the ELICODE$_{ALL}$ performance improves compared to the scores obtained by ELICODE on Czech, German, Italian and Swedish languages: training on both training and development set allows accounting for the similarities between development set and test set too. Consistently with the above-mentioned analysis, the performances achieved on the Swedish and Czech data sets are comparable and lower than the scores obtained on the German data set, that recorded the highest F0.5 score of 82.32%. Concerning the differences in the English data, as expected, ELICODE performs better than ELICODE$_{ALL}$ on both FCE and REALEC partitions, this is likely due to the high dissimilarity between the English FCE development and test data sets, thus training the model on the development set as well amounts to introducing noise during the learning phase. Additionally, given the great difference between FCE and REALEC partitions, the results of models trained on the FCE data set are consistently lower on REALEC data compared to the results on the FCE data.

In order to explore the impact of the difference between the English data sets, we trained a model only on the REALEC development set. The model has been trained for 10 epochs and by maintaining fixed all the other parameters so as to make the results of such model comparable to the others. The model trained only on REALEC data achieved 58.44 of Precision, 33.19 of Recall and the F0.5 is 50.72, thus improving the F0.5 of about 7% compared to the ELICODE result; in particular, the model becomes more precise in predicting errors, but given the reduced amount of training data is less incline to label tokens as incorrect.

Concerning the baseline, its poor performance is likely due to the employed representation: count-based features consider terms in isolation rather than in context, in so doing, the model is able to detect errors based on words frequency only, thus detecting errors related only to vocabulary—i.e. non-existing words or unseen tokens at training time. In this respect, the results achieved by the baseline on the REALEC partition of the English data set are lower than those for the FCE data set—especially on Precision—, thus reflecting the difference between such two data sets. Conversely, the representations employed by language models such as

XLM-RoBERTa are context sensitive—i.e. each token representation accounts for the whole sequence information—and this is reflected in a consistent improvement in Recall scores.

After this numerical analysis of the results, a more qualitative analysis of the errors of the system could have been useful to better understands the limits of the approach, but it was not possible because, to date, only the unlabelled test set were released on github.

In order to assess the multilingual competence of the language model, we trained a model on the concatenation of the training sets of all the different languages: typologically similar languages may mutually improve the model representations, while languages with different structures may negatively impact the error detection in both languages. The trained multilingual models, as said, follow the same experimental setting than the submitted monolingual models. Differently than the monolingual models which were trained for 10 epochs, the multilingual models have been trained for 7 epochs: in this setting the training took on average 55 hours per epoch for ELICODE$_{MLT}$ and 62 hours for ELICODE$_{MLT_{ALL}}$.[12]

The multilingual models perform similarly on the shared task test sets compared to monolingual models. If we consider the two languages with a smaller training and development sets, i.e. Italian and Swedish, we might notice that the performance on the Italian test set does not improve using the multilingual approach. This might be due to the fact that the other languages included in the shared task are not typologically similar to Italian. On the contrary, the performance on the Swedish language, which is slightly higher than the monolingual model performance, might benefit from the German training and development data sets, being both Germanic languages.

Giving now a quick look at the other systems that participated in the shared task, as reported in Volodina et al., 2023, the most successful approaches relied on BERT-like large language models: one employed mBERT (Brainstorm Thinkers, but no system description was submitted) and the other made use of XLM-RoBERTa like ours (DSL-MIM-HUS). Difference in performances between our system and the

[12]The multilingual model trained only on the training data sets (ELICODE$_{MLT}$) for 7 epochs achieved the same results of the 8-epoch model. Thus, we assume that ELICODE$_{MLT}$ reached the learning upper bound at the 7th epoch.

first one can be easily explained based on the differences in the employed model, while the second seems not to be so far from our approach: DSL-MIM-HUS, in fact, made use of XLM-RoBERTa by training it on all the languages at once, similarly to our ELICODE$_{MLT}$ and ELICODE$_{MLT_{ALL}}$ approaches. Differences in results may be due to training specificities (the Authors declared the use of a different library and 20 epochs of training). VLP-char and NTNU-TRH approaches, based on LSTM and GRU neural networks, showed the limits of these architectures compared to BERT-like models, both in monolingual and multilingual setting. The last model su-dali, only tested on the Swedish part of the dataset, training the generic transformers architecture by Vaswani et al., 2017 with artificial data mimicking the error distribution from the Swedish source corpus. They achieved very good precision results.

Chapter 4

Language models as Decoders

This Chapter shows a different perspective on language models: in particular, it is concerned with exploiting their generative features, on investigating how these can adapt to a certain type of language, and how these can be used to analyze language itself. This study, whose first steps were presented in (Colla et al., 2022), aims to understand if language models can support clinicians work at identifying people suffering from Alzheimer Disease. The investigation tests the reliability of the perplexity metric and its suitability to discriminate between impaired subjects and healthy controls. In this setting an English corpus, collected for research purposes was employed (Becker et al., 1994). This work has been extended to account for Italian, and for texts collected in more ecological fashion (as transcriptions of dialogues). For the purposes of this investigation an existing dataset, the Anchise corpus (Benvenuti et al., 2020), including transcript of conversation with impaired subjects, was complemented with a new corpus containing conversations transcripts where healthy elderly were involved.[1]

4.1 Semantic Coherence Markers: the Contribution of Perplexity Metrics

In economically developed societies the burden of mental disturbances is becoming more evident, with negative impact on people's daily life and huge cost for health systems. Whereas for many psychotic disorders no cures have been found yet, the treatment of people at high risk for developing schizophrenia or related psychotic

[1]A manuscript on this work is currently in preparation.

disorders is acknowledged to benefit from early detection and intervention (Marshall et al., 2005). To this end, a central role might be played by approaches aimed at analyzing thought and communication patterns in order to identify early symptoms of mental disorder (Larson, Walker, and Compton, 2010).

The analysis of human language has recently emerged as a research field that may be helpful to analyze for diagnosing and treating mental illnesses. In fact, in the last decade Natural Language Processing (NLP) techniques have become a common tool to support research on psychotic disorders. Namely, if language and its associated cognitive functions are first impaired before the full signs of mental disorders become apparent, linguistic analysis assisted by computing systems may be helpful for early detection. Recent advances in NLP technologies allow accurate language models (LMs) to be developed. These can be thought of as probability distributions over text sequences, and can be used to estimate in how far a text is coherent with (or, more precisely, predictable through) such language models. In order to measure the distance between an actual sequence of tokens and the probability distribution we propose using *perplexity*, a metric that is well-known in literature for the intrinsic evaluation of LMs. In this work we run experiments targeted at investigating how reliable perplexity is as a tool for investigating individuals' language, and we test whether the perplexity computed employing a language model acquired based on speeches from healthy subjects can be useful in discriminating healthy subjects from people suffering from mental disorders. While in previous reports the reliability of perplexity has been simply taken for granted, we investigate whether and to what extent perplexity scores are reliable before trying to use them to discriminate between mentally impaired and healthy subjects. Moreover we compare perplexity scores computed through LMs as diverse as GPT-2 and N-grams to the ends of discriminating healthy subjects from subjects afflicted by Alzheimer Disease.

4.1.1 Perplexity

As mentioned in Chapter 2, LMs are basically probability distributions of word sequences: perplexity was originally conceived as an intrinsic evaluation tool for LMs, in that it can be used to measure how likely a given input sequence is, given a LM (Goldberg, 2017). In other words it is a metric to evaluate the language model itself, not taking into account the specific tasks it's going to be used for. This measure is defined as follows. Let us consider a word sequence of k elements, $W = \{w_1, \ldots, w_k\}$; since we are interested in evaluating the model on unseen data, the test sequence W must be new, and not be part of the training set. Given the language model LM, we can compute the probability of the sentence W, that is $\mathrm{LM}(W)$. Such a probability would be a natural measure of the quality of the language model itself: the higher the probability, the better the model. The average log probability computed based on the model is defined as

$$\frac{1}{k} \log \prod_{i=1}^{k} \mathrm{LM}(W) = \frac{1}{k} \sum_{i=1}^{k} \log \mathrm{LM}(W),$$

which amounts to the log probability of the whole test sequence W, divided by the number of tokens in sequence. The perplexity of sequence W given the language model LM is computed as

$$\mathrm{PPL(LM,W)} = \exp\{-\frac{1}{k} \sum_{i=1}^{k} \log \mathrm{LM}(w_i|w_{1:i-1})\}. \tag{4.1}$$

It is now clear why low PPL values (corresponding to high probability values) indicate that the word sequence fits well to the model or, equivalently, that the model is able to predict that sequence.

From another point of view, perplexity can be considered as the weighted branching factor of the model. Let us consider a simple example: if we roll a regular fair die, with six faces. The weighted branching factor of the model is 6, simply indicating how many possible outcomes there are for each roll. Now, if we decide to roll an extremely unfair die, biased in order to return always 6, the weighted branching factor of the model would be 1: in fact, even if the branching factor would always be 6, because of the 6 faces, the only possible outcomes is 6, 1 face. Imagine now to train a model over these two dice and calculate the perplexity of the two models: in the first case the perplexity would be calculated knowing that every element of the sequence of rolls (for example 100 rolls), would be produced with a probability of $1/6$, causing a computed perplexity value of 6 (the same as the weighted branching factor). In the second case, considering again 100 rolls, the sequence produced by the model could be approximated with the probability of 99/100 for the face with 6 on it an 1/500 for every other face. The computed perplexity would be approximately 1, again the same as the weighted branching factor.

Following this idea and bringing back perplexity on the field of language modeling, we can interpret perplexity as the number of words on which the model is undecided to predict as the most probable. In other words, if we obtain an average perplexity of 20, it means that at every step of the way, the model considered more or less 20 words as the most probable over the entire vocabulary of words.

It is worth noticing that, as mentioned before, perplexity is normally used as an evaluation tool for language models themselves and the focus of the measure is on the comparison of values obtained by different models on the same inputs. Our use of the measure shift the focus on the input, measuring the capability of the measure itself to highlight differences or anomalies in input data, given a fixed model trained on specific types of text.

Importantly enough, perplexity is *corpus*-specific, that is it can be used to measure in how far a given LM (which has been acquired on a given training *corpus*) predicts an arbitrary word sequence.

4.1.2 Related work

Patients with psychiatric disorders such as schizophrenia show various semantic disturbances, and may suffer from difficulties in handling linguistic meanings at different processing levels such as morphology, syntax, semantics, and pragmatics (Boer et al., 2020). The work in (Covington et al., 2005) provides a rich overview on disturbances at the different levels. As far as we are concerned, disturbances related to schizophrenia typically produce abnormal usage of neologisms and word approximations, disruptions in language cohesion (Docherty, DeRosa, and Andreasen, 1996), syntactically simpler constructions featured by reduced use of embedded clauses and grammatical dependents (Çokal et al., 2018), inflectional morphology variants and errors (Walenski et al., 2010). In the last decade, advances in NLP techniques have allowed the construction of approaches to automatically deal with tasks such as linguistic analysis and production, including also many of the aforementioned linguistic levels. These approaches have identified markers that can help differentiate patients with psychiatric disorders from healthy controls, and predict the onset of psychiatric disturbances in high risk groups at the level of the individual patient.

Most approaches rely on a simple yet powerful descriptive (and predictive) linguistic device which is known as 'distributional hypothesis'.

The distributional hypothesis states that words that occur in similar contexts tend to convey similar meanings (Harris, 1954).

For example, if the word w_i and the word w_k often occur in the same context then probably w_i and w_k they probably have close meanings (and if they are interchangeable in the same contexts of occurrence, then they probably are synonyms). For example, in the sentences *We used the board to shut down the power plant.* and *We used the panel to shut down the power plant.* the words *board* and *panel* are intended with the same meaning.

Several techniques have been devised to acquire the distributional profiles of terms. The resulting vectors are hypothesized to be suitable for representing word meanings over a multidimensional Euclidean space, where distance acts like a proxy for similarity, and where similarity can be interpreted as a metric.

Early work in this area started with generating vectors from co-occurrence matrices (Harman, 1993; Schütze and Pedersen, 1997), treated with latent semantic indexing (Landauer, Foltz, and Laham, 1998), or point-wise mutual information (Hindle, 1990). Such early distributional representations provided *explicit* (that is, directly meaningful and human-interpretable) information. The number of dimensions of such vectors was determined by the size of the vocabulary. On the other side, in *implicit* or latent representations, features were used resulting from Latent Semantic Analysis (LSA). LSA is a multidimensional associative model based on the distributional hypothesis: word meaning is encoded as a multi-dimensional (usually 300 or 400 dimensions) vector obtained by elaborating large *corpora* to estimate the co-occurrence frequencies for each word. A basic approach based on LSA, such as that described in the seminal work by (Elvevåg et al., 2007), is as follows. Each input token is represented through a corresponding LSA vector, $W_i = \{I_{i1}, I_{i2}, \dots I_{iN}\}$. In turn, the vector representation for a phrase P is then built as the mean of the vectors representing all words in P: $P_i = \frac{1}{N}\sum_{k=1}^{N} I_{ik}$. The coherence between any two phrases is then computed through the cosine similarity of their corresponding vectors. The assumption underlying this approach is that meaningful texts will be featured by high coherence scores (in that words in the text being considered are semantically related on a distributional perspective), whilst text with some sort of disorder (or 'loose associations' among words) will be featured by reduced coherence scores. In (Bedi et al., 2015) an artificial dataset built by intentionally manipulating existing texts was used to test the described notion of coherence: the minimum semantic distance and the mean semantic distance of adjacent sentences were found to be negatively correlated with the disorder level introduced in the original. In this work LSA (in conjunction with information on grammatical Part-of-Speech function, referred to as POS tags) has been used to predict the transition to psychosis in a clinical high-risk cohort.

More recently, LSA techniques have been superseded by neural approaches aimed at learning latent representations of words called word embeddings (Navigli and Martelli, 2019). The overall design aimed at characterizing coherence (or, equivalently, the disorder associated with sentences and documents), by comparing vector representations of text excerpts, has remained unchanged. Among the most relevant sets of word embeddings, we mention Word2vec (Mikolov et al., 2013b), GloVe (Pennington, Socher, and Manning, 2014), ConceptNet Numberbatch (Speer and Chin, 2016), fastText (Bojanowski et al., 2017), LessLex (Colla, Mensa, and Radicioni, 2020), and NASARI (Camacho-Collados, Pilehvar, and Navigli, 2015).

A different approach to provide quantitative measures to language coherence and complexity is graph-based: in this setting, nodes represent words, and the word sequence is induced by directed edges. One main assumption underlying these approaches is that in coherent discourse neighboring words refer to connected topics, whilst incoherent discourse is associated with difficulties in making an ordered trajectory or path between topics. By employing tools from graph theory and information science it is possible to extract information on graph properties, such as connectedness, subgraphs or graph components. More specifically, measures such as entropy can be employed to probabilistically define topics and topic transitions (Cabana et al., 2011). Such graph representations also allowed grasping specific features of the normal and dysfunctional flow of thought (such as divergence and recurrence), and to produce accurate sorting of individuals affected by schizophrenia or mania (Mota et al., 2012). In another study, techniques for speech graph analysis were employed to describe formal thought disorder, which has been mathematically defined by the linear combination of connectedness graph attributes and their degree of similarity to randomly generated graphs. Such connectedness attributes were mapped onto a Disorganization Index, and used to classify negative symptom severity (Mota, Copelli, and Ribeiro, 2017).

In what follows we survey a set of works employing *perplexity* that are specifically relevant to introduce our own proposal. Although originally conceived to assess how language models are able to model previously unseen data, perplexity can be used to compare (and discriminate) text sequences produced by healthy subjects or by people suffering from language-related disturbances.

In (Stolcke and Shriberg, 1996) N-grams of part of speech (POS) tags were employed to identify patterns at the syntactic level. Then, two LMs were acquired (one from patients' data and the other from data from healthy controls): the categorization of a new, unseen (that is, not belonging to either set of training data) sample was then performed through the perplexity computed with the two LMs over the sample. The considered sample was then categorized as produced by a healthy subject (patient) if the LM acquired from healthy subjects (patients) data attained smaller perplexity than the other language model. Perplexity has been recently proposed as an indicator of cognitive deterioration (Frankenberg et al., 2019); more specifically, the content complexity in spoken language has been recorded in physiological aging and at the onset of Alzheimer's disease (AD) and mild cognitive impairment (MCI) on the basis of interview transcripts. LMs used in this research were built by exploiting 1-grams and 2-grams information; as illustrated in next section (please refer to Equation 2.2), such models differ in the amount of surrounding information employed. Perplexity scores were computed on ten-fold-cross-validation basis, whereby participants' transcripts were partitioned into ten parts; a model was then built by using nine parts and was tested on the tenth. This procedure was repeated ten times so that each portion of text was used exactly once as the test set. Four examination waves with an observation interval of more than 20 years were performed, and correlations of the perplexity score of transcriptions dating to the beginning of the experiment were found with the score from the dementia screening instrument in participants that lately developed MCI/AD.

Perplexity has been employed as a predictor for Alzheimer Disease (AD) on the analysis of transcriptions from DementiaBank's Pitt Corpus, that contains data from both healthy controls and AD patients (Becker et al., 1994). More precisely, in (Fritsch, Wankerl, and Nöth, 2019) two neural language models, based on LSTM models, were acquired, one built on the healthy controls and the other trained on patients belonging to the dementia group. A leave-one-speaker-out cross-validation was devised and, according to this setting, a language model $\mathcal{M}_{-s}$ was created for each speaker s by using all transcripts from the speaker's group but those of s. Data from speaker s was then tested on both $\mathcal{M}_{-s}$, thus providing a perplexity score p_{own}, and on the language model built upon the transcripts from the whole group to which the speaker did not belong to, thus obtaining the perplexity score p_{other}. The difference between the perplexity scores $\Delta_s = p_{own} - p_{other}$ was computed as a description for the speaker s. The classification of each speaker was then performed by setting a threshold ensuring that both groups obtained equal error rate. The authors achieved 85.6% accuracy on 499 transcriptions, and showed that perplexity can also be exploited to predict a patient's Mini-Mental State Examination (MMSE) scores. The approach adopted in this work is the closest to our own work we could find in literature; however it also differs from ours in some aspects. First, we investigated how reliable perplexity is in assessing the language of healthy subjects. That is, we analyzed how perplexity scores vary within the same individual, as an initial step toward assessing if perplexity is suitable for examining text excerpts/transcripts that (like in the case of the Pitt Corpus) were collected through multiple interviews and tests, spanning over years. Additionally, we were concerned with evaluating all excerpts from a single individual to predict the AD diagnosis at the subject level, rather than in predicting the class for each and every transcript. In order to assess the perplexity as a tool to support the diagnosis, we analyzed only data from subjects for which at least two transcripts were available.

Following the approach presented in (Fritsch, Wankerl, and Nöth, 2019), perplexity has been further investigated for the categorization of healthy subjects and

AD patients (Cohen and Pakhomov, 2020). In particular, different LMs have been acquired on both control and AD subjects' transcriptions from the Pitt Corpus (Becker et al., 1994). Such LMs have been employed to evaluate in how far differences in perplexity scores reflect deficits in language use. In order to compute perplexity scores, the authors designed two experimental settings: *interrogation by perturbation*, where LMs were asked to assess corrupted texts so to simulate AD progression; and *interrogation by interpolation*, where the perplexity values obtained by LMs acquired on healthy subjects transcripts were combined with perplexity values computed through LMs trained on the AD patients. In the classification task, the authors achieved their best results by assigning higher relevance to scores computed through LMs acquired on AD class rather than those trained on healthy subjects (AUC 0.941 and 0.872 accuracy at equal error rate). Also interestingly, the experimentation provided evidence about the correlation among perplexity scores and lexical frequency: provided that subjects affected by Dementia of the Alzheimer's type tend to use higher frequency words with less specificity than control individuals, language models and perplexity were proved to be able to capture linguistic manifestations of the cognitive impairment. Our approach differs from this one. Firstly, we explored two different sorts of LMs (N-grams and GPT-2 models, fine tuned with 5, 10, 20 and 30 epochs) so to collect experimental evidence on the level of accuracy recorded by different LMs used to compute the perplexity scores. Secondly, four different decision rules were compared based on average perplexity scores from control and impaired subjects, along with their respective standard deviations. Moreover, while in (Cohen and Pakhomov, 2020) the categorization is performed at the transcript level, our focus is on the categorization of subjects. More in general, our study is aimed at providing a full account on perplexity, and not only at investigating how to employ it in a categorization task.

4.1.3 Experiments

After having introduced the notion of perplexity and a brief description on modern neural architectures, we explore whether —and to what extent— the perplexity of LMs attained through such architectures can be used as a linguistic marker to detect language anomalies. Language anomalies detection may be helpful in recognizing mental disturbances and other disorders.

Before exploring perplexity as a tool suitable to discern linguistic anomalies in impaired subjects, we perform a preliminary step, consisting of checking whether perplexity scores can be considered reliable. Informally stated, by reliable we intend that similar text documents —such as repeated interviews to the same subject over a limited time span, or descriptions by different subjects about the same scene— should be featured by analogous perplexity scores (by employing the same language model). We designed two experiments, the first one aimed at exploring the intra-subject reliability of perplexity scores, and the second one aimed at exploring inter-subjects reliability. In the former case we recorded the coefficient of variation (CV) —that is the ratio between standard deviation and average perplexity scores—, and in the latter one we measured the intraclass correlation coefficients (ICC) (Shrout and Fleiss, 1979), that are two popular measures in the psychiatric psychometry community.

The whole experimentation presented in this Section is thus concerned with answering to two focal questions:

- whether perplexity scores are reliable within the same subject, but still sensitive enough to account for different sorts of speech forms produced by a given speaker (Experiment 1), and across subjects (Experiment 2);
- whether the language of a specific class of subjects, diagnosed as suffering from disorders impacting on common linguistic abilities, can be automatically distinguished from that of healthy controls solely based on perplexity accounts (Experiment 3).

In the first experiment we analyzed whether the LMs acquired by training both N-grams and GPT-2 on transcriptions of two different kinds of speech (two classes: political rallies *vs.* interviews) from a single subject produce different perplexity scores when the LM is used for analyzing similar (taken from same class) and different (from the other class) documents. In the second experiment we have measured the perplexity scores featuring discourses by 8 well-known political figures: in this case our aim was to assess whether perplexity scores computed based on models acquired on the other 7 speakers' transcripts are coherent in assessing the eight speaker. Finally, for the third experiment we have used the Pitt Corpus, from which we selected the transcripts of responses to the Cookie Theft stimulus picture (Goodglass and Kaplan, 1983), and investigated whether the perplexity score allows discriminating patients with dementia diagnosis (n = 194) from healthy controls (n = 99).

The code for replicating the experiments is available at `https://github.com/davidecolla/semantic_coherence_markers`.

Compared LMs

Three different experimental setups have been designed in order to compare perplexity as computed by language models acquired by training with two different sorts of architectures: N-grams, and GPT-2.

N-grams Since N-grams implement the simplest language model with context, where each word is conditioned on the preceding N-1 tokens only, we adopted N-grams for the first experimental setup. For the sake of clarity we introduce the formalization for Bigrams; such formulation can be further generalized to any N. We define the probability of a sequence of words $W_{1,n} = \{w_1, w_2, \dots, w_n\}$ as:

$$P(W_{1,n}) = \prod_{i=1}^{n} P(w_i|w_{i-1})$$

where the probability of each Bigram is estimated by exploiting the Maximum Likelihood Estimation (MLE) (Jurafsky and Martin, 2014, Chap. 3).[2] According to the MLE, we can estimate probability of the Bigram (w_{i-1}, w_i) as:

$$P(w_i|w_{i-1}) = \frac{C(w_i|w_{i-1})}{C(w_{i-1})} \tag{4.2}$$

where $C(w_i|w_{i-1})$ is the number of occurrences of the Bigram (w_{i-1}, w_i) in the training set, while $C(w_{i-1})$ counts the occurrences of the word w_{i-1} only. It is worth mentioning that training Bigrams on a limited vocabulary may lead to cases of

[2]In this setting, stopwords are customarily not filtered, as providing useful sequential information.

out-of-vocabulary words, i.e., unseen words during the training process. Out-of-vocabulary words pose a problem in calculating the probability of the sentence in which they are involved: in such cases we are not able to compute the probability of the Bigram involving the unknown word, thus undermining the probability of the whole sequence. In order to deal with out-of-vocabulary words, each token occurring only once in the training set can be replaced with the 'unknown' tag, UNK. In so doing, during the test phase we are allowed to map each out-of-vocabulary word to the unknown word tag. Of course this procedure entails that the probability mass associated to UNK tokens tends to overestimate the role of such tokens, badly affecting the behavior of N-gram based models. Conversely, modern architectures such as GPT-2 are less impacted from out-of-vocabulary (OOV) issues: in fact, such models are acquired by employing huge amounts of data (in the order of 40 GB of text (Radford et al., 2019)) by using sub-word tokenizers and encoding strategies.

Notwithstanding the strategy for handling out-of-vocabulary words, we may still end up with unseen N-grams, formally occurring zero times in the training set, thereby resulting in a null probability. We addressed the unseen N-grams issue through the interpolated Kneser-Ney Smoothing technique (Kneser and Ney, 1995). The most effective smoothing techniques for N-grams involve exploiting lower-order representations so to improve the precision of higher-order N-grams whenever is needed. For example if the 3-gram (w_{i-2}, w_{i-1}, w_i) has zero evidence, we may either rely solely on the probability of its lower-order components, that are the bigrams (w_{i-2}, w_{i-1}), (w_{i-1}, w_i) and the unigrams (w_i), (w_{i-1}), (w_{i-2}) or combine the scores of its lower-order components to obtain the higher-order representation. The Kneser-Ney algorithm belongs to the family of interpolation strategies, and is based on the absolute discounting technique: to compute a precise probability distribution, we may need to *discount* the counts for frequent N-grams to save some probability mass to deal with unseen N-grams: in so doing, we subtract a small discounting factor d from the counts of N-grams to employ such discount as probability for unseen N-grams.

In the present setting we experimented with N-grams ranging from 2- to 5-grams; the Kneser-Ney discounting factor d was set to 0.1.[3]

The vocabulary was closed on each experiment: that is, the N-grams models employed in each experiment were acquired with the vocabulary obtained from the concatenation of the transcripts herein. Since the perplexity is bounded by the vocabulary size, fixing the cardinality of the vocabulary allows obtaining comparable perplexity scores from N-gram models trained across different corpora.

GPT-2 The second experimental setup that we designed exploits the GPT-2 neural model, in particular we used the GPT-2 pre-trained model available via the Hugging Face Transformers library.[4] In this setting, the input text has been preprocessed by the pre-trained tokenizer and grouped into blocks of 1024 tokens. The pre-trained model is specialized as Causal Language Model (CLM) on the input texts, that is, predicting a word given its left context. Since the average log-likelihood for each token is returned as the loss of the model, the perplexity of a text is computed according to Equation 4.1.

4.1.4 Experiment 1: Intra-subject and discourse-level coherence

The first experiment is aimed at investigating whether perplexity scores computed based on a given LM are stable, and whether perplexity scores are able to grasp factors specific to a given sort of speech. We have then targeted transcripts of two different kinds of discourse: the interview and the political rally. While in the former case both the questions put to the interviewee and his answers may be featured by different topics political rallies are events where people sharing similar political beliefs gather to support their candidate, whose language is in principle more regular, as not concerned with answering to specific questions. As regards as the linguistic register differentiating such transcripts, interviews should convey a sense of poise, balance, and posture, while the language adopted in rallies is expected to be more emphatic, direct, uniform and vehement. Our second research question was then

[3]To compute N-grams we exploited the Language Modeling Module (*lm*) package from NLTK version 3.6.1, https://www.nltk.org/api/nltk.lm.html.

[4]https://huggingface.co/gpt2

TABLE 4.1: Statistics describing the transcripts employed in Experiment 1: for all considered samples we report time duration, number of tokens, number of unique tokens, average number of tokens and of unique tokens, and type-token ratio (TTR).

Category	Transcript	Duration	Tokens	Unique Tokens	AVG Tokens	AVG Unique Tokens	TTR
Interview	I	1 : 28 : 52	7,278	1,098	8,953	1,185	0.13
	II	1 : 28 : 23	6,471	922			
	III	1 : 31 : 34	18,514	1,926			
	IV	0 : 45 : 40	6,702	1,032			
	V	1 : 01 : 51	5,933	946			
Rally	I	1 : 17 : 37	15,200	1,967	15,051	1,944	0.13
	II	0 : 56 : 17	10,501	1,614			
	III	1 : 43 : 43	20,865	2,300			
	IV	1 : 13 : 01	14,056	1,945			
	V	1 : 18 : 19	14,806	1,896			

whether the employed language models were able to recognize the two different linguistic registers.

Materials We selected 10 transcripts by the former US President Donald Trump (this choice is mostly due to the large availability of his transcripts): 5 interviews and 5 campaign rallies were downloaded from the Rev platform.[5] Interviews were recorded between June 2019 and November 2020, while campaign rallies date to September and October 2020. The duration of both interviews and rallies varies between 45 minutes and one hour and 43 minutes. The statistics describing all transcripts employed in the first experimental setting, including time duration, token counts and type-token ratio (TTR, computed as the ratio between the types, that is the total number of different tokens[6] occurring in a text divided by the total number of tokens) are reported in Table 4.1. While the initial choice of the transcripts was random within each category, we tried to select text excerpts of similar duration.

Procedure Two types of model were acquired, one for Political Rallies and one for Interviews, and this schema was replicated for both N-grams and GPT-2. Each LM was then tested on *leave-one-out* basis on transcripts in the same category as the training/fine-tuning, and in direct fashion on transcripts from the other category. In the following we will simply refer to training, even though in a strict sense

[5]`https://www.rev.com`.
[6]Tokens are intended as proper words, not subwords as output of a tokenizer.

training procedures were employed to acquire N-gram models, while fine-tuning[7] is associated to the refinement step of the base GPT-2 model (more on fine-tuning in Church, Chen, and Ma, 2021). For example, in order to compute the perplexity score for excerpts from the Rally category with a language model obtained by training/fine-tuning on the same category, 5 models were built by using 4 of the 5 available transcripts (the fifth one was used for testing); results were then averaged over these 5 runs. Conversely, to compute the perplexity score on excerpts from the Interview category one LM was acquired from the Rally class, and used to test on all 5 transcripts. The same procedure was followed for the training/fine-tuning on the Interview category: leave-one-out schema for testing on transcripts from the same class, and only one model to compute the perplexity of transcripts in the other class.

As regards as the LMs acquired through GPT-2, the selected transcripts were employed for the fine-tuning. Provided that most systems cited in related literature adopt 20 epochs (an epoch being the hyperparameter that governs the number of complete runs all throughout the training dataset), we explored if either it is possible to obtain similar results with models tuned with less epochs, or higher categorization through further fine-tuning epochs. We thus experimented with 5, 10, 20 and 30 epochs, together with the pre-trained model. In order to compute the perplexity we adopted a *sliding window* of 50 tokens; window sizing was motivated by the fact that on average, the sentence length for transcripts is around 50 words (namely, 53.63 for interviews and 51.67 for rallies).

We then expected to observe analogous perplexity scores on all transcripts (as capturing common features underlying the language of the same speaker); and to observe slightly higher perplexity scores with models trained/fine-tuned on Interviews (Rallies) and used to test on Rallies (Interviews). In order to assess the reliability of PPL scores, we recorded the coefficient of variation (CV), which is computed as the ratio between standard deviation of the perplexity scores and the average

[7]Our distinction is compatible with a definition provided in literature: "In fine-tuning, we begin with off-the-shelf embeddings like word2vec, and continue training them on the small target corpus" (Jurafsky and Martin, 2014, p.399). Another popular description of the goals of fine-tuning comes from the work in (Devlin et al., 2018): "During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the [. . .] model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters".

TABLE 4.2: Perplexity (PPL) scores along with standard deviation scores obtained with fine-tuning on the transcripts from the Rally and Interview categories, averaged values for perplexity (PPL) scores, standard deviations and coefficient of variation (CV). The top four rows illustrate the results on the N-gram models, while the bottom rows show the results obtained by employing GPT-2 models, varying from 0 (pre-trained model) to 30 fine-tuning epochs.

Model		Rally						Interview					
		Rally			Interview			Rally			Interview		
		avg-PPL	avg-stdev	CV	avg-PPL	avg-stdev	CV	avg-PPL	avg-stdev	CV	avg-PPL	avg-stdev	CV
N-grams	2-gr	296.81	2.71	0.01	304.18	15.78	0.05	282.77	4.19	0.01	270.35	13.17	0.05
	3-gr	525.59	8.86	0.02	545.30	33.25	0.06	489.08	11.02	0.02	457.84	26.05	0.06
	4-gr	709.04	11.89	0.02	730.75	43.77	0.06	645.93	14.49	0.02	592.29	33.68	0.06
	5-gr	914.90	16.94	0.02	931.04	70.41	0.08	817.71	19.84	0.02	734.16	50.72	0.07
GPT-2	0 ep	27.16	1.20	0.04	24.15	1.55	0.06	27.16	1.20	0.04	24.15	1.55	0.06
	5 ep	18.29	0.66	0.04	17.64	1.21	0.07	20.22	1.00	0.05	18.44	1.34	0.07
	10 ep	16.36	0.55	0.03	17.02	1.35	0.08	18.59	1.01	0.05	17.22	1.19	0.07
	20 ep	15.16	0.51	0.03	17.16	1.61	0.09	18.18	1.06	0.06	16.80	1.12	0.07
	30 ep	14.86	0.50	0.03	18.05	2.02	0.11	18.50	1.14	0.06	17.12	1.19	0.07

of perplexity scores. A CV $\leq .1$ would contribute to support the hypothesis that perplexity provides stable and reliable scores.

Results The results are presented in Table 4.2, where we recorded the average perplexity scores, their standard deviation and the coefficient of variation.

We observe that PPL scores grow monotonically from 2-grams to 5-grams; as regards as GPT-2 based models, PPL scores tend to decrease from the base model up to 20 fine tuning epochs, while they grow when computed though models acquired by 30 training epochs. In this case also standard deviation grows, which means that such models are overfitting to the fine-tuning data. The coefficient of variation is on average lower than 0.1 (we recorded .06 CV on the scores from GPT-2-based models, and .04 for the scores from N-gram models).

As regards as our second research question, whether perplexity allows recognizing different linguistic registers, we recorded a twofold result. In fact, while PPL scores acquired from rallies show reduced CV scores when tested on transcripts from the same class (Table 4.2), models acquired on interviews provide lower CV values when tested on rallies. This result may be explained to some extent with the observation that data available to train/fine-tune such models were roughly half of data available for rallies. We defer to future work a deeper investigation and experimentation on this point.

Experiment 2: Inter-subject coherence on different speakers

The second experiment was aimed at assessing whether perplexity scores are stable across subjects. Five transcripts with no specific topic for eight well-known past and present political figures were selected, and a language model for each subject was trained/fine tuned. The perplexity score was then computed for the speeches from each speaker, based on the others' language models (thus 7 LMs were used to compute the PPL scores for each one of the 8 speakers). In this case we expected to record analogous PPL scores by employing the models trained on the other speakers: a good agreement through models trained on different speakers would support the reliability of the PPL metrics.

Materials In this case the context was less uniform than in the previous experiment, in that we collected political rallies, speeches on spot topics, such as economy, health systems, general challenges for the Western economy, a talk given in the frame of the World Economic Forum in Davos (Switzerland), civil rights, and so forth. Statistics describing time duration, number of tokens, number of unique tokens and type-token ratio describing the transcripts employed in this experiment are presented in the Appendix, in Table A.1.

Procedure A speaker *vs.* speaker setup was implemented, that is all transcripts for each subject were employed to fine-tune a GPT-2 model or to acquire N-grams. The models obtained from each subject were then used to compute perplexity scores for the transcripts from other subjects.

Similar to the former experiment, in all experimental conditions involving language models based on GPT-2 we compared results obtained through models refined with 5, 10, 20 and 30 fine-tuning epochs, with a sliding window sized to 50 tokens.

The transcripts of each speaker were 'rated' (with PPL scores) through the models acquired from the other seven speakers. The set of ratings collected for all speakers were then compared, to investigate to what extent the series of PPL scores can be deemed as reliable. To explore the reliability of the perplexity scores we employed the Intraclass correlation coefficients (ICC) (Shrout and Fleiss, 1979). In this setting, ICC values above 0.9 are recognized to indicate excellent reliability (Liljequist, Elfving, and Skavberg Roaldsen, 2019). Six ICC variants may be overall considered, according to the choice of raters, and to whether a single measurement or the average of 2 or more measurements are employed (Shrout and Fleiss, 1979), so that ICC models are featured by two parameters, as in ICC(X,Y). The former variable specifies the model, that is how raters are chosen. Model 1: each subject is assessed by a different set of randomly chosen raters; model 2: each subject is assessed by each rater, and raters are randomly sampled; model 3: each subject is assessed by each rater, and the set of raters is fixed. The second variable reports whether reliability should be computed based on a single measurement, or by employing the average of 2 or more measurements provided by different raters. We therefore chose the ICC(3,1) metric, that is each subject was assessed by all raters, and the set of raters kept constant (as indicated by the first argument, '3'); also, a single measurement was employed (as indicated by the second argument, '1').

Results The detailed PPL scores and standard deviations recorded in the second experiment are presented in Table A.2 in Appendix Sextion A.2, which reports figures averaged over the 5 transcripts available for each subject. In Table 4.3 we provide the ICC scores obtained from those runs. The ICC scores show high (> 0.8) correlation for N-gram based models, and very high correlation (> 0.9) for GPT-2 based models. Thus we obtained good to optimal reliability for perplexity scores computed through the models at stake. As regards as N-gram models, the implementation employing closed dictionary over all subjects obtained substantially increased reliability scores with respect to the naïve implementation employing a dictionary closed given a single speaker.

By inspecting the results obtained by both GPT-2 and N-grams-based models, we observe high ICC scores, that reduce as long as fine-tuning proceeds. This trend may be explained by noticing that when we extend fine-tuning, language models tend to be less general and to over-fit the language of an individual speaker, thereby becoming progressively less able to account for the language of all other ones. On the whole, these scores show that different GPT-2 based models do provide reliable PPL scores when used to assess the speeches of individual subjects. In this task there

TABLE 4.3: Intraclass correlation coefficients characterizing the perplexity scores obtained in the second experiment, in which each speaker was rated through LMs acquired/fine tuned based on transcripts from all other speakers.

Model	ICC(3,1) score
2-grams	0.88
3-grams	0.86
4-grams	0.83
5-grams	0.80
GPT2-5	0.98
GPT2-10	0.97
GPT2-20	0.94
GPT2-30	0.91

is no need for biasing models towards a specific subject's language, and fine tuning turns out to be detrimental to the reliability of PPL scores.

Experiment 3: Predictive and discriminative features of PPL

For this experiment we used publicly available data from the Pitt Corpus.[8] These data were gathered as part of a larger protocol administered by the Alzheimer and Related Dementias Study at the University of Pittsburgh School of Medicine (Becker et al., 1994). In particular, we selected the descriptions provided to the Cookie Theft picture, which is a popular test used by speech-language pathologists to assess expository discourse in subjects with disorders such as dementia. This experiment was designed to investigate whether perplexity scores on the collected descriptions allow discriminating patients from healthy controls.

Materials The dataset is composed of 552 files arranged into Control (243 items) and Dementia (309 items) directories. These correspond to multiple interviews to 99 control subjects, and to 219 subjects with dementia diagnosis. Text documents herein were transcribed according to the CHAT format,[9] so we pre-processed such documents to extract text. In so doing, the original text was to some extent simplified: e.g., pauses were disregarded, like hesitation phenomena, that were not consistently annotated (MacWhinney, 2014; MacWhinney, 2017).

[8]https://dementia.talkbank.org/access/English/Pitt.html.
[9]https://talkbank.org/manuals/CHAT.pdf.

```
*INV:   what I want you to do is look at the picture and just tell me
        anything you see going on .
%mor:   pro:int|what pro:sub|I v|want pro:per|you inf|to v|do cop|be&3S
        v|look prep|at det:art|the n|picture coord|and adv|just v|tell
        pro:obj|me pro:indef|anything pro:per|you v|see n:gerund|go-PRESP
        adv|on .
%gra:   1|3|LINK 2|3|SUBJ 3|0|ROOT 4|3|OBJ 5|6|INF 6|3|COMP 7|6|OBJ 8|7|CPRED
        9|8|JCT 10|11|DET 11|9|POBJ 12|8|CONJ 13|14|JCT 14|12|COORD 15|14|OBJ
        16|14|OBJ 17|18|SUBJ 18|14|COMP 19|18|OBJ 20|18|JCT 21|3|PUNCT
*PAR:   well the kids are in the kitchen with their mother &uh &uh takin(g)
        cookies out o(f) the cookie jar .
%mor:   co|well det:art|the n|kid-PL cop|be&PRES prep|in det:art|the
        n|kitchen prep|with det:poss|their n|mother part|take-PRESP
        n|cookie-PL adv|out prep|of det:art|the n|cookie n|jar .
%gra:   1|4|COM 2|3|DET 3|4|SUBJ 4|0|ROOT 5|4|JCT 6|7|DET 7|5|POBJ 8|4|JCT
        9|10|DET 10|8|POBJ 11|4|XJCT 12|11|OBJ 13|12|NJCT 14|13|JCT 15|17|DET
        16|17|MOD 17|14|POBJ 18|4|PUNCT
```

FIGURE 4.1: Excerpt from the Pitt Corpus: first interview with the subject #6 of the control group encoded in the CHAT format. The lines beginning with *INV and *PAR refer to the transcriptions for *Investigator* and *Participant*, respectively. Lines starting with %mor and %gra report both morphological and grammatical analysis of the transcript line. Interjections such as "uh" are marked with &, while incomplete words such as "takin" are completed in the transcript as "takin(g)".

In Figure 4.1 we illustrate an excerpt, encoded in the CHAT format, taken from the Pitt Corpus. The transcriptions of the subject were selected —i.e., we retrieved only the lines starting with *PAR—, discarding the texts from the investigator. The CHAT transcription format is very rich and informative; for example, incomplete words are completed in a post-processing stage and marked through brackets, "bro" is represented as "bro(ther)"; pauses of the speaker are marked through dots in brackets, for example (.) indicates a short pause while (...) refers to a longer pause; interjections are marked with the symbol &, for example "&uh" or "&ehm". Such elements were discarded; experiments exploiting this sort of information were left for future work. In particular lengthened syllables, long pauses and interruption symbols were eliminated, alongside a wide variety of sounds such as cries, sneezes, and coughs. Other meaningful aspects were preserved in the final file, such as repetitions, interjections and retracings, considering these events as important features for the model to capture. No information on intonational contours and other markers of the utterance planning process was available in the input files.

To the ends of collecting enough text to be analyzed, we dropped the interviews of subjects that participated in only one interview. We ended up with material relative to 74 control subjects (for which overall 218 transcripts were collected), and to

TABLE 4.4: Statistics describing the transcripts employed in Experiment 3. For each class we report the average number of tokens per interview, the average number of unique tokens per interview, the number of participants, the overall number of transcripts

and the type-token ratio (TTR).e

Class	AVG Tokens	AVG Unique Tokens	Participants	Transcripts	TTR
Control	437	26	74	218	0.07
Alzheimer's Disease	409	25	77	192	0.08

77 subjects with dementia diagnosis (overall 192 transcripts).

The statistics describing number of tokens, number of unique tokens and type-token ratio for the transcripts employed in the Experiment 3 are presented in Table 4.4.

Procedure This experiment is aimed at testing the discriminative features of perplexity scores: more specifically, we tested a simple categorization algorithm to discriminate between mentally impaired and healthy subjects. We adopted the experimental setup from the work in (Fritsch, Wankerl, and Nöth, 2019): two language models LM_C and LM_{AD} were acquired by employing all transcripts from Control and Alzheimer's disease groups, respectively. Such models are supposed to grasp the main linguistic traits of both groups speeches, thus representing the typical language adopted by subjects belonging to Control and AD classes. For both groups we adopted a leave-one-subject-out setting, whereby language models were fine-tuned with files from all other subjects within the same group except for one, which was used for testing. For each subject s we acquired the model LM_s on the transcripts from the same group of s, except for those of the subject s. Each transcript in the corpus was then characterized by two perplexity scores P_C and P_{AD}, expressing the scores obtained through language models acquired on Control and AD groups, respectively. More precisely, if a subject s was a member of the AD class, the scores P_C for its transcripts were obtained through LM_C, while the scores P_{AD} were computed by exploiting LM_s. Vice versa, if the subject s was from the Control group, the scores P_C for her/his transcripts were obtained through LM_s, while the scores P_{AD} were computed by exploiting LM_{AD}. Additionally, since we were interested in studying the scores featuring each subject, we synthesized the perplexity scores P_C and P_{AD} of each subject with the average of her/his transcripts scores, thus obtaining $\overline{P}_C$ and

$\overline{P}_{AD}$.

In order to discriminate AD patients from healthy subjects, we adopted a threshold-based classification strategy. Three different approaches were explored to estimate such threshold:

(i) in the first setting we used the average perplexity scores characterizing all control subjects employed in the training process ($\overline{P}_C$);

(ii) in the second setting we computed the threshold as the average perplexity score of all the subjects belonging to the AD class ($\overline{P}_{AD}$);

(iii) in the third setting we estimated two different thresholds by exploiting the difference $\overline{P}_{AD} - \overline{P}_C$, by initially following the approach reported in (Fritsch, Wankerl, and Nöth, 2019) and (Cohen and Pakhomov, 2020).

For each subject, the threshold estimation process was computed through a leave-one-subject-out setting, and repeated for the three approaches from (i) to (iii). In the first setting the threshold was estimated on all the subjects from the control group except for the test subject s: for each subject s we computed the threshold as the average of $\overline{P}_C$ scores for all subjects in the control group except for s —if s was from the healthy controls group— (Figure 4.2). In case the perplexity score $\overline{P}_C$ for the subject s was higher than the healthy controls threshold, we marked the subject as suffering from AD; as healthy otherwise.

Similarly, in the second setting we computed the threshold as the average of $\overline{P}_{AD}$ scores for all subjects in the AD group except for s. In case the perplexity score $\overline{P}_{AD}$ for the subject s was higher than the average of AD class threshold, we marked the subject as healthy; as suffering from AD otherwise (Figure 4.3).

The rationale underlying the first two settings is that each subject may be characterized more accurately by LMs acquired on transcript from the same group: in other words, we expected lower perplexity scores to be associated to control (AD) subjects, rather than subjects belonging to the other class, with LMs trained or fine-tuned on transcripts from control (AD) subjects.

Following the literature, in the third setting we characterized each subject with the difference $D = \overline{P}_{AD} - \overline{P}_C$. We defined two thresholds, $\overline{D}_{AD}$ which was computed

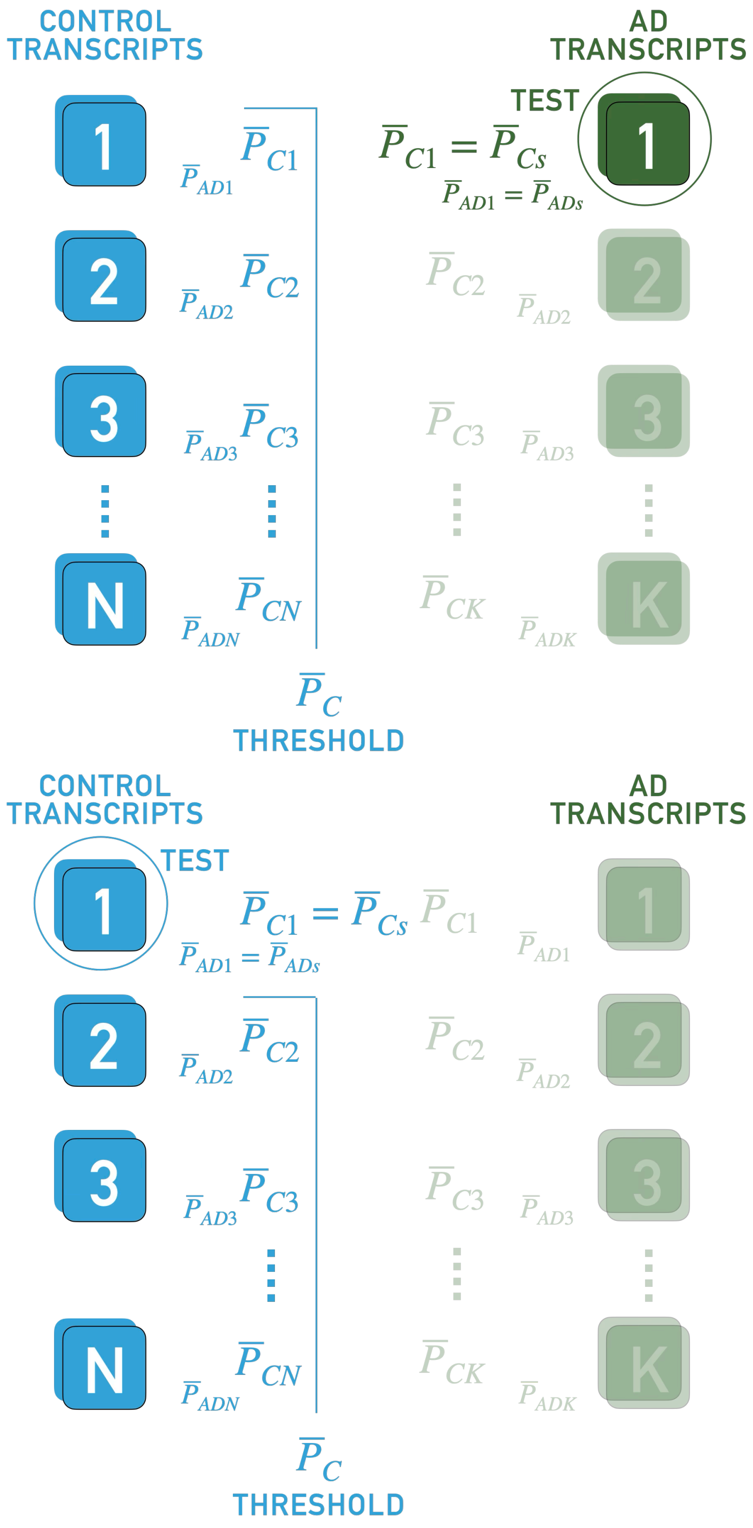


FIGURE 4.2: The first two rules are quite similar: In both cases a threshold was calculated as mean of the perplexity values of every subject of the corresponding class, $\overline{P}_C$ or $\overline{P}_{AD}$ ($\overline{P}_C$ in the reported example). Following the leave-one-out-rule, when the test subject was out of the group that was used to calculate the threshold, the $\overline{P}$ value could be calculated as mean all the subjects in the group. When the test subject was instead part of the group we calculated the threshold from, it was left out when calculating the threshold value.

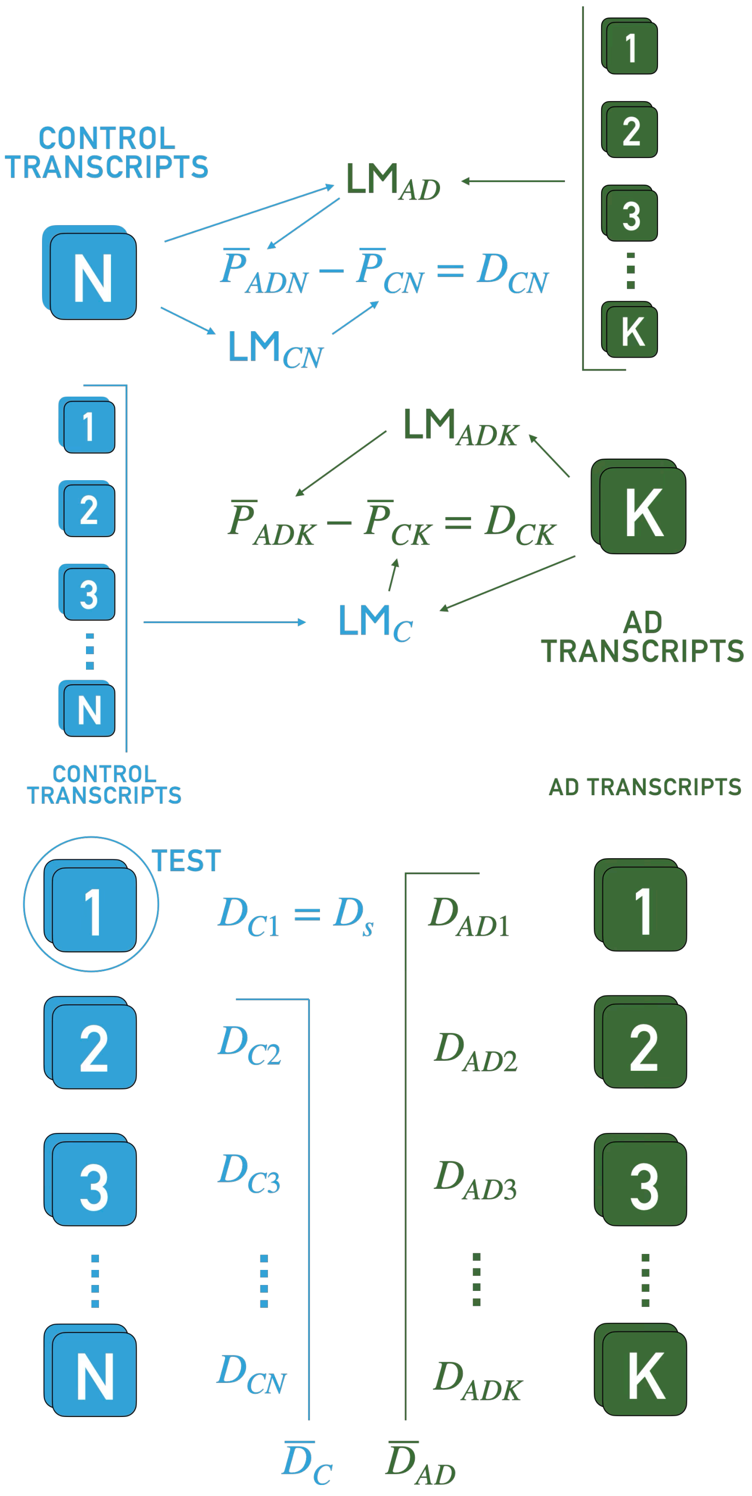


FIGURE 4.3: Using rule (iii), every subject would not be defined anymore by two value, but one: the two perplexity scores are produced as before with the language models associated to C and AD group, but now we will take in consideration the difference of the two values.The result is a value $\overline{D}_{CN}$ for the Nth control group subject and $\overline{D}_{ADK}$ for AD subjects. We categorize a subject S by choosing the class associated to the threshold (either $\overline{P}_{AD}$ or $\overline{P}_C$) featured by smallest margin with the D_S score of the subject.

as the average of all the difference scores from patients in the AD group and $\overline{D}_C$, defined as the average of all the difference scores from healthy controls. In both cases we considered all the patients belonging to the group except for the test subject s (s was held out with the only purpose to rule out her/his contribution from $\overline{D}_{AD}$ or $\overline{D}_C$).

Different from literature —where equal error rate is used—, we employ $\overline{D}_{AD}$ and $\overline{D}_C$ as compact descriptors for the classes AD and C, respectively. The rationale underlying this categorization schema is that a subject is associated to the class that exhibits most similar perplexity score to her/his own. We categorize a subject s by choosing the class associated to the threshold (either $\overline{D}_{AD}$ or $\overline{D}_C$) featured by smallest margin with the PPL score computed based on a given LM for the transcripts from s, T_s according to the following formula:

$$\text{class}(s) = \underset{x \in \{C, AD\}}{\text{argmin}} \left| D - \overline{D}_x \right| . \tag{4.3}$$

This setting (involving $\overline{D}_{AD}$ and $\overline{D}_C$) will be referred to as $\overline{D}$.

Furthermore, we refined the decision rule $\overline{D}$ to account for standard deviation information. Together with the average $\overline{D}_{AD}$ and $\overline{D}_C$, we computed also σ_{AD} and σ_C as the standard deviations of the difference scores D for impaired and control groups. We explored the 3σ rule, which is a popular heuristic in empirical sciences: it states that in populations that are assumed to be described by a normally distributed random variable, over 99.7% values lie within three standard deviations of the mean, 95.5% within two standard deviations, and 68.3% within one standard deviation (Helms, 2009). On this basis we explored the three options by adding 1, 2 and 3 standard deviations to average scores: the best results were obtained by employing 2 standard deviations. Our thresholds were then refined as follows:

$$\begin{aligned} \overline{D}^*_{AD} &= \overline{D}_{AD} + 2 \cdot \sigma_{AD}, \text{ and} \\ \overline{D}^*_C &= \overline{D}_C - 2 \cdot \sigma_C . \end{aligned}$$

The updated decision rule for categorization was then reshaped as

$$\text{class}(s) = \underset{x \in \{C,AD\}}{\text{argmin}} \left| D - \overline{D}^*_x \right| . \tag{4.4}$$

This setting, involving $\overline{D}^*_{AD}$ and $\overline{D}^*_C$, will be referred to as $\overline{D}^*$.

A twofold experimental setting has been devised, including experiments with N-grams and GPT-2, adopting a window size set to 20 in order to handle shorter text samples (the shortest text in the training data contains only 23 tokens). In the case of N-grams, the models were acquired for 2-grams to 5-grams; the GPT-2 model was fine-tuned employing 5, 10, 20 and 30 epochs.

Evaluation Metrics To evaluate the results we adopted the Precision and Recall metrics (specificity and sensitivity) along with their harmonic mean, F1 score, and accuracy. Precision (specificity) is defined as $P = \frac{TP}{TP+FP}$, while Recall (sensitivity) is defined as $R = \frac{TP}{TP+FN}$. Informally stated, Precision computes the fraction of results that are actually correct: it is computed as the number of correct results (true positives, TP) divided by the sum of correct results (TP) and items mistakenly returned as results (false positives, FP). Recall computes how many correct results were individuated. In Recall, we have the number of correct results divided by the sum of correct results (TP) and items mistakenly not recognized as results (false negative, FN). While precision provides an estimation of how precise a categorization system is, recall indicates how many results were identified out of all the possible ones. F_1 measure is then used to provide a synthetic value of Precision and Recall, whereby the two measures are evenly weighted through their harmonic mean: $F_1 = 2 \cdot \frac{P \cdot R}{P+R}$

Accuracy was computed as $ACC = \frac{TP+TN}{P+N}$, that is as the fraction of correct predictions (the sum of TP and TN) over the total number of records examined (the sum of positives and negatives, P and N).

Finally, in order to record a synthetic index to assess accuracy and F1 scores on the two groups at stake, we used the harmonic mean among these three values. It was computed as

$$\text{HM}(\text{Acc.}, \text{F1}_{AD}, \text{F1}_C) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{x_i} \tag{4.5}$$

where n was set to the number of x_i values being averaged.

Results The overall accuracy scores are presented in Figure 4.4, while detailed figures across different experimental conditions are presented in Table A.4, in A.4.

Let us start by reporting the results from N-gram models. The overall most effective strategy is $\overline{D}^{*}$ (Eq. 4.4), based on a threshold using the difference between AD patients and healthy controls, extended with the 3σ rule. The best performing model is based on Bigrams, and obtained .93 accuracy, .92 F1 score on the AD class, and .93 F1 score on the C class. The models employing PPL scores from the control group (indicated as P_C in Figure 4.4 and in Table A.4) obtained the lowest accuracy scores in all conditions, well below the random guess, while the accuracy yielded by the $\overline{P}_{AD}$ strategy is always above .5.

In general we observe that increasing the length of the Markovian assumption reduces the accuracy of N-gram models for all decision rules (employing more context seems to be slightly detrimental for such models), with the exception of the $\overline{D}$ strategy.

The results obtained by the GPT-2 models reveal overall higher accuracy, ranging from .71 for the best model acquired with 5 epochs of fine-tuning to 1.00 for all further fine-tuning steps. The same profile describes the F1 scores recorded on the sub-tasks focused on AD and control subjects, respectively, varying from around 0.69 for the best model acquired with 5 epochs of fine-tuning ($\overline{D}$ strategy on the AD class) to 1.00 for all other models and sub-tasks. If we consider the efficacy of thresholding strategies and associated decision rules, the refined difference rule $\overline{D}$ is the best performing strategy for GTP-2 based models, as witnessed by the rightmost column in Table A.4. Such scores report the harmonic mean among accuracy, F1 score on categorization of AD subjects and on categorization of control subjects. A compact view on data from the same column is provided in Table 4.5, illustrating the best strategy for each model at stake.

To frame our results with respect to literature, let us start from the accuracy of the baseline clinical diagnosis obtained in the first version of the study by Becker et al., 1994: it was 86%, and after considering follow-up clinical data this datum raised to 91.4%, with a 0.988 sensivity and 0.983 specificity. This is what subsequent literature considered as the gold standard against which to compare experimental

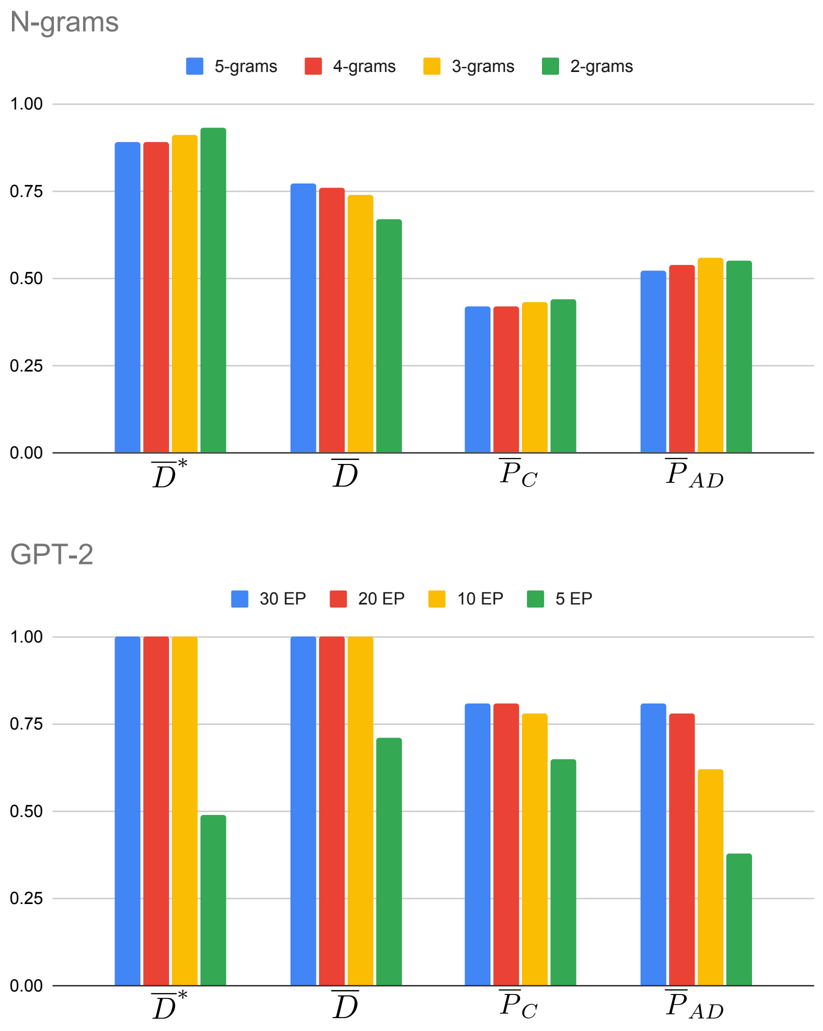


FIGURE 4.4: Plot of the accuracy scores for the third experiment on the categorization of AD/control subjects. The histograms in the top sub-figure show the accuracy on N-grams, while the histograms at the bottom report results obtained through GPT-2 models. Different colors correspond to N-gram of differing order and to different fine-tuning epochs, respectively. The histograms illustrate the scores obtained through $\overline{D}^*$, $\overline{D}$, $\overline{P}_C$ and $\overline{P}_{AD}$ decision rules, respectively.

TABLE 4.5: Study to compare the effectiveness of the thresholding and categorization strategies for each LM. The top scoring strategy is reported for each model.

N-gram models	categorization strategy	mean HM score
2-grams	$\overline{D}^*$	0.93
3-grams	$\overline{D}^*$	0.91
4-grams	$\overline{D}^*$	0.89
5-grams	$\overline{D}^*$	0.89
GPT-2 models: epochs	categorization strategy	mean HM score
5 epochs	$\overline{D}$	0.71
10 epochs	$\overline{D}, \overline{D}^*$	1.00
20 epochs	$\overline{D}, \overline{D}^*$	1.00
30 epochs	$\overline{D}, \overline{D}^*$	1.00

outputs. We recall that such data are particularly relevant as human evaluation included various analytical steps, such as medical and neurologic history and examination, semistructured psychiatric interview, and neuropsychological assessments. Experimental results provided in subsequent work approach those ratings by employing solely transcripts of descriptions to a rather simple picture. A relevant work attained 85.6% accuracy through LSTM based models (Fritsch, Wankerl, and Nöth, 2019) in the categorization of individual transcripts. Such results were then replicated and improved in the work by (Cohen and Pakhomov, 2020), where the best reported model experimentally obtained a 0.872 accuracy.

General Discussion Provided that our experimental results seem to outperform the accuracy scores reported in literature, we feel that one main relevant result of this experimentation is that evidence was provided that perplexity scores are reliable at both intra-subject and inter-subject levels, and suited to categorize the language of subjects affected by cognitive impairments. In doing so, only speakers' transcripts were used.

Additionally, we realized that a short, controlled elicitation task can potentially outperform natural linguistic data obtained from speakers. The quality of our results needs be checked in different settings (further languages, varied experimental conditions: much experimental work still needs to be done), but this fact provides evidence that specialists may be effectively assisted by systems employing a technology based on language models and perplexity scores. Also, by comparing language models as different as N-grams and models based on the more recent GPT-2, we observed that Bigrams outperform a GPT-2 model fine tuned for 5 epochs. This fact may provide insights on the possible trade-off between accuracy of the results and computation time and costs.

While perplexity proved to be overall a viable tool to investigate human language, we found consistent differences in the outputs of the models at stake, mostly stemming from intrinsic properties of the LMs, from the amount of context considered by the models, from the size of available training data, and from the amount of training employed to refine models themselves. One first datum is that even though N-grams can be hardly compared to GPT-2-based models, nonetheless it may be helpful trying to discern the scenarios in which such models provide better results. In Experiment 1, which can be considered as a rather favorable experimental setting for N-gram models, recorded CV scores are on par or smaller than those obtained through GPT-2 based models. In Experiment 2 the ICC scores characterizing N-gram-based model output (ranging from 0.88 to 0.80) show valuable reliability. As anticipated, this is probably the most challenging setting for N-grams, in that the samples are featured by a consistent number of unique tokens and nearly doubled TTR with respect to documents employed in Experiment 1. Also, selected documents span various topics and a significant time frame, going from the mid Sixties to 2020. It was somehow surprising, then, that in Experiment 3 the accuracy level attained by the best-performing N-gram model (2-grams) achieved a 0.93 harmonic mean improving on the best GPT-2-based model fine tuned for 5 epochs and employing the $\overline{D}$ decision rule (HM=0.71; please refer to Table A.4).

This result may be understood in the light of the rather regular language used for the descriptions to the Cookie Theft picture, that thereby turned out to be less demanding for the N-gram LMs. In these respects, a lesson learned is that N-grams can be employed in scenarios where the task is less difficult on lexical and linguistic accounts (recorded TTR values roughly range on 0.08, 0.25 and 0.13 for the Experiments 1, 2 and 3, respectively): in some instances of such problems adopting N-gram models may be convenient (considering both training and testing efforts) with respect to the more complete and computationally expensive Transformer models. Few data may be useful to complete this note on the trade-off between accuracy and computational effort. Our experiments were performed on machinery provided by the Competence Centre for Scientific Computing (Aldinucci et al., 2017). In particular, we exploited nodes with 2x Intel Xeon Processor E5-2680 v3 and 128GB memory. The first experiment took on average 12 hours for each GPT-2 LM, and about 5 minutes for all the N-gram models. The second experiment lasted about 32 hours for each GPT-2 LM and about 12 minutes for all the N-gram models, while the third experiment took around 8 hours for each GPT-2 setting and about 12 minutes for all the N-gram models.

4.2 Recognizing Dementia Spectrum Disorders through Plain Conversation Transcripts Analysis

We then moved to experimenting on the same categorization task as before (healthy *vs.* cognitively impaired) in Italian, and also to introduce a more ecological scenario for the conversations being considered. To carry out the experimentation we employed an existing corpus (the Anchise Corpus (Benvenuti et al., 2020)) which was complemented by the newly collected Nestor Corpus. The Nestor Corpus contains conversations with healthy subjects, while the Anchise Corpus only contains (transcripts of) conversations with impaired subjects. Two different experiments were thus conducted:

1. **PPL-based categorization.** In the first experiment we tried to automatically categorize the set of transcripts by employing the perplexity metric, following the idea we already adopted in the previous Section, using N-grams and an italian version of GPT-2, `GePpeTto`, (De Mattei et al., 2020).

2. **Human Categorization.** In the second experiment we measured to what extent humans are able to discriminate between healthy and cognitively impaired subjects.

4.2.1 Datasets: Anchise corpus and Nestor corpus

Anchise Since 2007, healthcare professionals affiliated with the Anchise Group have recorded, transcribed, and annotated conversations involving elderly individuals experiencing cognitive impairments, primarily residing in Italian Nursing Homes. Specifically, participants include Italian speakers exhibiting evident cognitive deficits (as per inclusion criterion: MMSE score = [0-28]), with no psychological or behavioral disturbances impeding communication, such as drowsiness, wandering, marked oppositional or aggressive demeanor, noticeable psychomotor agitation, severe dysarthria, or profound hypoacusia. The Mini-Mental State Examination (MMSE) score, reported for each patient, as reported in Folstein, Folstein, and McHugh, 1975, is a simplified, scored form of the cognitive mental status assessment, that concentrates only on the cognitive aspects of mental functions.

The Anchise Corpus was initially collected to serve as instructional material for Nursing Home staff training sessions, to train healthcare worker to follow the 'enabling approach' (more on such approach in the following), which is intended to promote the wellness of dementia patients through plain conversations. Continuously updated by the Anchise Group, a previous release of the corpus, the "Corpus Anchise 320" (Vigorelli, Benvenuti, and Bolioli, 1994; Benvenuti et al., 2020), containing 320 conversations, was released in 2020. Since then the corpus was further updated with new conversation transcripts, and the Anchise 2022 Corpus encompasses 417 conversations. For the sake of the present experimentation we considered the subset of the transcripts provided with a MMSE score ≤ 25 resulting in a set of 234 records.

The adoption of the ApproccioCapacitante® ("Enabling Approach") is central to the methodology of the Anchise Group, pioneered by one of the authors and their collaborators since 2004 (Vigorelli, 2004), which has evolved over subsequent years (Vigorelli, 2010), (Vigorelli, 2011); (Lanzoni et al., 2018); (Vigorelli, 2021)). This approach is a cornerstone of non-pharmacological therapies for dementia, and is aimed at facilitating patient communication, particularly those impaired by memory and language deficiencies, with the overarching objective of fostering harmonious interactions among individuals. Integral to the implementation of this approach is the engagement in conversations between Individuals Living With Dementia (ILWD) and trained professionals of various disciplines (including healthcare workers, educators, nurses, speech therapists, doctors, and psychologists) who have undergone specific training in the Enabling Approach.

Nestor Another dataset, the Nestor corpus, was compiled consisting of conversations with healthy elderly subjects. We recruited participants over 70+ years old, with a Mini-Mental State Examination (MMSE) score of 26 or higher (indicating cognitive normalcy). All subjects volunteered for the study.[10]This corpus contains transcripts from conversations with 28 healthy elder subjects. The conversation collection procedure mirrored the Enabling Approach employed for the Anchise corpus.

[10]Ethical approval was granted by the Research Ethics Board (Comitato di Bioetica di Ateneo) of the University of Turin (Protocol number 0303348, June 12th, 2023). The study adhered to the ethical principles of the Declaration of Helsinki for Human Research (World Medical Association). Authors were anonymized throughout data collection.

Detailed statistics for the dataset, including total tokens, unique tokens, and type-token ratio (TTR; calculated as the ratio of unique tokens to total tokens), are provided in Table 4.7.

Table 4.6 provides a rough comparison between some descriptive statistics of the employed corpora. Compared to the Anchise, the Nestor corpus it is considerably smaller, with subjects being on average slightly younger. The gender ratio is less skewed towards females than in Anchise, and subjects boast near-maximum MMSE scores (29.4 out of 30). Interestingly, both the interviewed subjects and operators contribute a higher average number of conversation turns in the Nestor corpus, while turn length remains comparable between the two corpora. Consequently, conversations within Nestor are significantly longer on average, despite both corpora exhibit substantial standard deviation.

TABLE 4.6: Descriptive statistics for the Anchise corpus and the Nestor corpus; we report the size of both corpora, some statistics describing subjects involved in the collected conversations, a rough analysis of speech turns composing conversations and of conversations themselves. The length of speech turns and conversations is expressed as the number of tokens therein.

	Filtered Anchise	**Nestor**
Number of conversations	234	28
Average subject age (stdev)	84.06 (6.98)	77.71 (6.14)
Male subjects (percentage)	51 (22%)	13 (46%)
Female subjects (percentage)	183 (78%)	15 (54%)
AVG subject MMSE (stdev)	12.55 (5.40)	29.41 (1.12)
AVG num. of turns per conv.	65.12 (42.11)	90.29 (60.07)
AVG num. of subject turns per conv.	32.19 (20.86)	43.71 (30.01)
AVG num. of operator turns per conv.	32.91 (21.31)	42.50 (30.44)
AVG turn length (stdev)	12.36 (10.73)	12.96 (4.89)
AVG turn length Subject (stdev)	17.25 (20.72)	18.88 (10.08)
AVG turn length Operator (stdev)	7.41 (3.29)	7.97 (2.22)
AVG conv. length (stdev)	700.56 (467.59)	1122.07 (806.13)
AVG conv. length - Subject (stdev)	460.76 (355.65)	783.96 (632.20)
AVG conv. length - Operator (stdev)	239.81 (182.71)	338.11 (248.64)

4.2.2 Experiment 1: PPL-based categorization

With this new corpus we decided to test the discriminative features of perplexity scores in this new condition, using an ecological text as reflected by the "real world" setting.

TABLE 4.7: Statistics describing transcripts from Nestor dataset. For each patient we report the total number of tokens, the number of unique tokens, and the type-token ratio (TTR).

Subject ID	Tokens	Unique Tokens	TTR
1	459	222	0.48
2	984	389	0.40
3	1314	521	0.40
4	782	341	0.44
5	547	238	0.44
6	316	164	0.52
7	1265	528	0.42
8	234	153	0.65
9	575	255	0.44
10	263	142	0.54
11	244	142	0.58
12	668	312	0.47
13	546	254	0.47
14	314	186	0.59
15	866	358	0.41
16	560	248	0.44
17	1159	431	0.37
18	308	172	0.56
19	461	227	0.49
20	1443	551	0.38
21	599	275	0.46
22	579	287	0.50
23	484	240	0.50
24	537	277	0.52
25	633	275	0.43
26	2199	702	0.32
27	511	256	0.50
28	3101	894	0.29
Average	783.96	322.86	0.46

Procedure

As before, we utilized the same experimental setup from the study by (Fritsch, Wankerl, and Nöth, 2019): two language models LM_N and LM_A were obtained by utilizing all transcripts from Nestor and Filtered Anchise groups, respectively. These models aim to capture the primary linguistic characteristics of both groups' speeches, thereby representing the typical language employed by subjects belonging to Nestor and Anchise classes. The methodology we followed is the same we described in experiment 3, but for the sake of clarity we prefer to cast the explanation on Anchise and Nestor corpora.

For each group, we employed a leave-one-subject-out approach, where language models were fine-tuned with files from all other subjects within the same group except for one, which was reserved for testing. For each subject s, we obtained the model LM_s using transcripts from the same group as s, excluding those of subject s. Each transcript from both groups was then assessed with two perplexity scores P_N and P_A, representing the scores obtained through language models acquired on Nestor and Anchise groups, respectively. In this manner, all transcripts (234 from the original Anchise corpus and the 28 from the Nestor corpus) were utilized in the experimentation and testing.

More specifically, if a subject s belonged to the Anchise class, the score P_N for their transcript was obtained using LM_N, while the score P_A was computed using LM_s. Conversely, if the subject s was from the Nestor class, the score P_N for their transcript was obtained using LM_s, while the score P_A was computed using LM_A. Each subject s was then characterized by a *score D*:

$$D = \mathrm{P_A} - \mathrm{P_N}. \tag{4.6}$$

Following exactly rule (iii) of the previous Section, we established two thresholds, $\overline{D}_A$, calculated as the average of all the *difference* scores from patients in the Anchise corpus, and $\overline{D}_N$, defined as the average of all the *difference* scores from subjects in the Nestor corpus. In both cases, we considered all subjects belonging to the group except for the test subject s (with s excluded solely to prevent their contribution from affecting $\overline{D}_A$ or $\overline{D}_N$). In contrast to literature, where the equal error rate is typically used, we utilize $\overline{D}_A$ and $\overline{D}_N$ as concise descriptors for the classes A and N, respectively. The rationale behind this classification approach is that a subject is associated with the class exhibiting the most similar perplexity score to their own. We assign a category to a subject s by selecting the class associated with the threshold (either $\overline{D}_A$ or $\overline{D}_N$) that has the smallest margin with the previously computed *score*, according to the following formula, the same we saw in Equation 4.3:

$$\mathrm{class}(s) = \underset{x \in \{C,AD\}}{\mathrm{argmin}} \left| D - \overline{D}_x \right|. \tag{4.7}$$

This setting, involving $\overline{D}_A$ and $\overline{D}_N$, will be referred to as $\overline{D}$.

TABLE 4.8: PPL-based categorization results according to each model employed.

Model	Pred. Task	Acc.	P	R	F1	HM(Acc., F1(P_D), F1(P_{HC}))
GPT-2	P_D	99.6%	99.6%	100%	99.8%	99.2%
	P_{HC}		100%	96.4%	98.2%	
2-Grams	P_D	56.9%	92.3%	56.4%	70%	39.9%
	P_{HC}		14.3%	60.7%	23.1%	
3-Grams	P_D	68.3%	95.8%	67.5%	79.2%	52.6%
	P_{HC}		21.6%	75%	33.6%	
4-Grams	P_D	72.5%	97.1%	71.4%	82.3%	58.2%
	P_{HC}		25.6%	82.1%	39%	
5-Grams	P_D	75.2%	97.7%	73.9%	84.2%	61.6%
	P_{HC}		28.2%	85.7%	42.5%	

We designed a twofold experimental setup, involving experiments with N-grams and GPT-2, with a window size set to 20 to accommodate shorter text samples. For N-grams, the models were trained for 2-grams to 5-grams; the GPT-2 model underwent fine-tuning over 10 epochs (this choice was made based on the results described in the previous Section, where 10 epochs seemed to represent the upper bound for the performance of our model). For all the details of the used language model, the description is exactly the same the one for the previous experiment, reported in Section 4.1.3.

Results

The results obtained with the experimental setups presented above are provided in Table 4.8. The experimental outcomes attained with various setups exhibit considerable diversity. The `GePpeTto` model stands out with the most outstanding performance, boasting a notable Harmonic Mean (HM, computed in Equation 4.5) of 99.2%, signifying a high level of proficiency in subject categorization. Conversely, N-Grams models generally yielded significantly lower Harmonic Means, indicating a relatively restricted capacity. The 2-Grams model secured an HM of 39.9%. Progressing to higher orders, noticeable enhancements were observed. The 3-Grams model achieved an HM of 52.6%, followed by the 4-Grams model with an HM of 58.2%, and the 5-Grams model reaching an HM of 61.6%. These findings suggest that as the order of N-Grams increases, there is a corresponding improvement in performance, emphasizing the influence of model complexity on task proficiency.

Comparing these results with those of Section 4.1, we can observe that N-grams seem to behave worse than they did in the previous setting, while GPT-model confirms good success rates in grasping language peculiarities. This is maybe due to the fact that in the setting described in Section 4.1 the transcripts from the Pitt Corpus were not as ecological as those in the Anchise and Nestor corpora, with a more controlled environment (all subjects were describing the Cookie Theft Picture) and less variability in terms of linguistic register, topics and vocabulary. N-grams reasonably were more affected by these changes, while the accuracy obtained through GPT-based models seems to remain unaltered.

However, the disparity in performance between different models, naturally entails costs, both in terms of time and computational resources. `GePpeTto` models require substantial computing power and time commitment for both training and perplexity computation. Specifically, for model training alone, the server used for our work required approximately 1 hour and 50 minutes for each model, totaling 264 models—consisting of 234 + 28 models in a leave-one-subject-out setting for each subject from both corpora, along with 2 global models, one for each corpus. This amounts to roughly 20 days of computational effort. In contrast, N-Gram models entail significantly lower resource requirements. On the same server and for the same number of models, considering both training and perplexity computation for all orders (ranging from 2-Grams to 5-Grams), only around 2 hours of computational time were needed.

Furthermore, even in terms of disk space consumption, GPT-2 models are far more resource-intensive compared to N-Gram models. Considering the same number of trained models, the GPT-2 folder comprises approximately 113GB of data, whereas the N-Grams folder contains only about 67MB.

4.2.3 Experiment 2: human categorization

Procedure

We ran a preliminary experiment in which two different teams of raters were recruited: a team was composed by psychologists, whilst the other one included raters with no background in either psychological or medical disciplines. The assignment proposed to our raters was formulated as follows: "The following transcripts include 14 dialogues with elderly individuals aged 75 years or older. The lines of the dialogues are marked as 'S' (speaker) and 'O' (operator): 'S' turns refer to the speakers, while 'O' indicates the interviewer. Some speakers have cognitive impairments, while others are healthy. The task is to evaluate whether the individuals involved in these dialogues are healthy or not. You are requested to mark each dialogue indicating whether the involved speaker is healthy or impaired."

The accuracy obtained by human raters, along with the inter-rater agreement in the experts group, is an interesting datum to assess the overall difficulty of the task (for both experts and non-experts), and to elaborate on the social perception of cognitive deficit, based on purely linguistic evidence. Results in this experiment may be also compared to those obtained in a seminal work on Alzheimer disease by Becker et al., 1994; this work, specifically relevant to computational approaches, provided an initial 86.0% baseline accuracy in the clinical diagnosis; after considering follow-up clinical data, this datum raised to 91.4%.

The two groups of raters were asked to evaluate the conversations of 14 subjects, 7 of which belonging to the Anchise corpus and 7 to the Nestor corpus.

Discussion

Raters	Pred. Task	Acc.	P	R	F1	HM(Acc., F1(P_D), F1(P_{HC}))
Non-experts (6 tests)	P_D	73.8%	76.2%	73.8%	74%	73.6%
	P_{HC}		74.2%	73.8%	73%	
Experts (5 tests)	P_D	87.2%	84.3%	91.4%	87.6%	87.1%
	P_{HC}		90.9%	82.8%	86.6%	

TABLE 4.9: Human annotators results, averaged on a pool of 6 non-experts and 5 experts, on a test set composed by 14 subjects, 7 from each corpus.

The Anchise corpus contains a widely varied set of cognitive impairments (as witnessed by MMSE scores obtained by the interviewed subjects, ranging in [0, 25]) which is expected to characterize this categorization task as a challenging assignment. In Figure 4.5 is provided an insightful about MMSE scores distributions among both datasets.

Also, dealing with spontaneous speech in an ecological setting involves facing possibly different linguistic registers and topics that may be harder with respect to

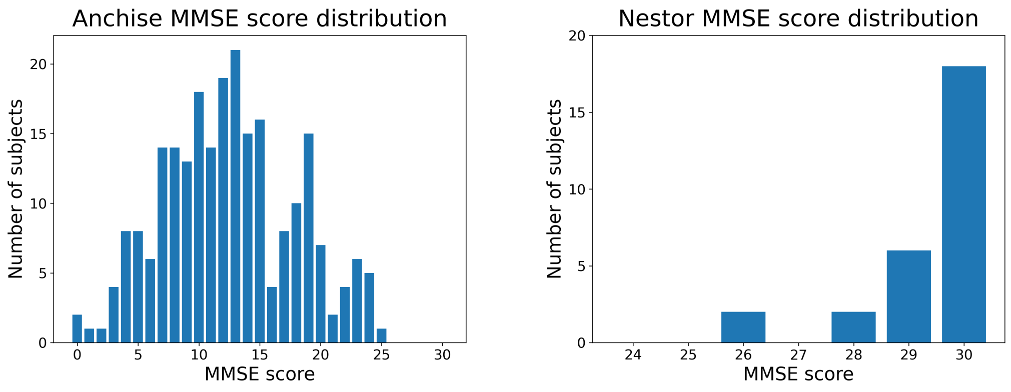


FIGURE 4.5: Distribution of the MMSE scores featuring the subjects from the the Anchise and Nestor corpus.

more controlled linguistic corpora (e.g., the transcripts of responses to the Cookie Theft stimulus picture in the Pitt Corpus (Goodglass and Kaplan, 1983)), where subjects were all solicited to describe the same scene.

Chapter 5

SE-MaCaROoN

This chapter is the final element of this exploration of Language Models capabilities. Unlike the previous ones, this chapter investigates how to use language for a foundational task in Lexical Semantics, Word Sense Disambiguation (WSD): exploiting both Embeddings generation and Natural Language generation capabilities, we obtained a novel lexical resource named SE-MACAROON (after Sense Embeddings from MAny ContextuAl RepresentatiONs), a collection of contextualized vectors. These vectors are created by combining information from WordNet (Miller, 1995) and contextual word embeddings generated by a language model, demonstrating how adding a semantic layer on top of a language model can improve its performance on tasks like WSD. The overall approach also exploits the generative capabilities of language models, that are employed to enforce a controlled data augmentation step, targeted at acquiring a bag of contextualized representations.

5.1 Preliminaries

While Language Models and Language Modelling task have been introduced in Chapter 2, in Chapter 3 the encoder module has been employed for information retrieval and extraction purposes, in Chapter 4 an application of the decoder module has been showed, in the present one we introduce the work on a novel Lexical resource.

When talking about lexical resources we refer to both semantic networks, using vertices of a labeled graph as representation of senses —introduced in Section 5.1.1— , and to distributional resources, representing word senses through dense vectorial representations, introduced in Section 5.1.2.

5.1.1 Wordnet

WordNet, a manually curated English word sense inventory developed by experts, is maybe the most popular resource of the first kind (Miller, 1995). Word senses are represented through synsets, sets of synonym terms expressing distinct word meanings. Each lemma (be it a single word or multi-word expression) in WordNet belongs to one or more synsets, and word senses are represented as a combination of word form (lexicalization) and synset (often referred to as sense-key). These nodes are uniquely identified by a synset id, and are endowed with a gloss and optionally by usage examples. Let us consider the word *bank* as an example. Depending on the surrounding context, amongst other senses it might refer to either the river bank or the financial institution. The river bank word sense is defined as *sloping land (especially the slope beside a body of water)*, with the synset containing only the term *bank* and identified by the synset ID `wn$09213565$n`. Additional information includes two examples: *they pulled the canoe up on the bank* and *he sat on the bank of the river and watched the currents*. The financial institution word sense is represented by the synset *depository financial institution, banking concern, banking company, bank*, identified by the ID `wn$08420278$n` and defined as *a financial institution that accepts deposits and channels the money into lending activities*. The reported usage examples are *he cashed a check at the bank* and *that bank holds the mortgage on my home*.[1]

WordNet is further categorized into four parts, each hosting a different part of speech: nouns, verbs, adjectives, and adverbs. These categories have their own internal relations. Nouns are organized hierarchically in a lexical memory, verbs are linked through various entailment relations, and adjectives and adverbs are positioned within N-dimensional hyperspaces. Each of these lexical structures reflects a unique way of capturing human experience.

The current version of WordNet (3.0) contains 117,659 synsets, 206,949 senses (sensekeys), and 147,306 distinct lemmas. Inspired by WordNet, researchers have invested considerable effort in developing other semantic networks and translating WordNet into various languages (Pianta, Bentivogli, and Girardi, 2002; Bond and

[1]We refer to WordNet v. 3.0, available at `https://wordnet.princeton.edu/download/current-version`

Paik, 2012; Rudnicka, Witkowski, and Kaliński, 2015; McCrae, Wood, and Hicks, 2017; McCrae et al., 2020).

5.1.2 Word and Sense Embeddings

The second type of lexical resources rely on the distributional hypothesis that words appearing in similar contexts often share similar meanings (Harris, 1954). For instance, if words w_i and w_k frequently co-occur in the same contexts, they are likely to have close meanings. In some cases, if they are interchangeable within these contexts, they may be considered as synonyms. Various techniques have been devised to capture the distributional patterns of words, typically using dense real-valued vectors in a high-dimensional space. Each word is represented by a vector, often called a "word embedding" and these vectors are positioned in a multidimensional space where distance (e.g., Euclidean distance) reflects semantic similarity. In simpler terms, words with similar meanings tend to be closer together in this space, while words with different meanings are positioned farther apart.

The most widely known and used embeddings are Word2vec (Mikolov et al., 2013b), GloVe (Pennington, Socher, and Manning, 2014), ConceptNet Numberbatch (Speer and Chin, 2016). Word embeddings ensure high correlation with human ratings on semantic similarity —in the order of .8 Spearman's ρ correlation (Speer and Lowry-Duda, 2017; Goikoetxea, Soroa, and Agirre, 2018), with some differences based on the considered datasets—, but they are opaque, in that they do not allow to specify which senses underlie the similarity score. All such vectors are term- rather than sense-indexed: regardless of the technique employed for the embedding generation, this sort of representation inherently suffers from a significant drawback: it ignores the fact that words can have multiple meanings and conflates all these meanings into a single representation (Camacho-Collados and Pilehvar, 2018). However, this limitation, known as meaning conflation deficiency, can hinder the ability of an NLP system relying on word embeddings to accurately grasp the intended meaning of the text: word embeddings seem to be unable to grasp different meanings of a word, even if those meanings occur in the training corpus (Schütze, 1998; Yaghoobzadeh and Schütze, 2016). Additionally, the deficiency may affect the semantic modeling of word senses, for example two semantically unrelated words similar to different word senses of the same word may be pulled together (Neelakantan et al., 2014; Pilehvar and Collier, 2016).

In order to alleviate the meaning conflation deficiency of word embeddings, a parallel direction of research has emerged over the past years, which tries to directly model individual meanings of words. In this Section we focus on sense representations relying on word embeddings, starting from the partition of word embeddings in *static* and *contextualized* representations. In fact, we now need to introduce the distinction of sense representations according to the underlying word embeddings design paradigm: either sense- or term-oriented. Additionally, attempting to address the meaning conflation limitation, mainly two lines of sense vector representations have emerged: in the first line, unsupervised, senses are learned directly from text corpora or in some knowledge-based fashion (Reisinger and Mooney, 2010; Huang et al., 2012; Vu and Parker, 2016; Vilnis and McCallum, 2014); while in the second approach senses are linked to pre-defined sense inventories. In this work we focus on the latter type of representation.

Static Sense Embeddings

Provided that the evolution of word representations flows from static word embeddings to contextualized word embeddings, the first vector representations for word senses have been introduced relying on static word embeddings. One of the main contributions between the sense vectorial representations is NASARI (Camacho-Collados, Pilehvar, and Navigli, 2015; Pilehvar and Navigli, 2015). In the same spirit of BabelNet, NASARI puts together two sorts of knowledge: one coming from WordNet (originally handcrafted by a team of lexicographers), based on synsets and on the intervening semantic relations, and one available in Wikipedia, which is conversely the outcome of a large collaborative effort. Pages in Wikipedia are considered as concepts. In NASARI embeddings each item is defined through a dense vector over a 300-dimensional space. NASARI vectors have been acquired by starting from the vectors trained on the Google News dataset, provided along with the Word2vec toolkit. All NASARI vectors share the same semantic space also with

Word2vec (Mikolov et al., 2013a), so that their representations can be used to compute semantic distances between any two such vectors. Thanks to the structure provided by the BabelNet resource, the resulting 2.9M embeddings are part of a huge semantic network. NASARI includes sense descriptions for nouns, but not for other grammatical categories.

SENSEEMBED was introduced directly by following NASARI, but with totally different building rationale, and containing representations for the four main parts of speech (nouns, verbs, adjectives and adverbs) (Iacobacci, Pilehvar, and Navigli, 2015). The approach proposed by SENSEEMBED is aimed at obtaining continuous representations of individual senses. In order to build sense representations, the authors exploited Babelfy (Moro, Raganato, and Navigli, 2014) to disambiguate the September-2014 dump of the English Wikipedia.[2] Subsequently, the Word2vec toolkit has been employed to build vectors for 2.5 millions of unique word senses. The obtained resource contains the representation for both terms —e.g., the embedding for the term Bank— and word senses —e.g., the embedding representing the meaning of bank intended as *financial institution*, endowed with the identifier Bank-bn:00008364n—
.

Given the negative impact of the meaning conflation deficiency implicitly coded in word embeddings, DECONF has been introduced with particular attention to the mentioned limitation (Pilehvar and Collier, 2016). DECONF is a sense representation technique that starts from a semantic network and a set of pre-trained word embeddings. The proposed approach computes a list of "sense biasing words" for a given word sense. The whole process is characterised by two phases: *(i)* the extraction of the most representative words that express the semantics of a synset, and *(ii)* the sense representations learning. In the extraction phase, the control strategy starts from a target synset y_t, leverages the structure of the semantic network of WordNet and produces as output an ordered list $\mathcal{B}_t$ of semantically related terms that provide a cue for the sense usage. The latter phase is aimed at learning the representation of the word sense s_t (the sense for term t): to these ends the procedure deconflates the representation of all the lexicalizations of the sense s_t, and biases them towards the list $\mathcal{B}_t$. In order to generate the DECONF resource, the authors chose WordNet 3.0 as

[2]http://dumps.wikimedia.org/enwiki/.

semantic network and the 300-*d* Word2Vec word embeddings trained on the Google News dataset. The final resource contains about 207 thousand vectors for WordNet word senses, each such sense representation lives in the same space which is also shared by the word embeddings.

Different from previous resources, SW2V (so named after 'Senses and Words to Vectors') is a neural model devised to represent both term and sense vector representations (Mancini et al., 2017). The proposed approach jointly learns both representations by exploiting text corpora and semantic networks. Due to the temporal complexity of the state-of-the-art disambiguation systems, the authors devised an unsupervised shallow word sense connectivity algorithm. Such algorithm exploits the connections of a semantic network and associates a term with its top candidate senses according to the number of sense connections and word context. Once the corpus of sense tagged words has been generated, an extension of the Word2Vec's CBOW model (Mikolov et al., 2013a) is employed. The extension of the CBOW model in order to deal with word senses follows the assumption that since a word is a lexicalization of an underlying sense, an update of the word embedding should entail a similar update of the sense representation, and vice versa. The authors chose BabelNet as reference semantic network and its underlying sense inventory; the pre-trained version of SW2V contains over 6 million vectors representing both words and word senses, in the same spirit of SENSEEMBED.

Following SW2V, LSTMEMBED is a recently proposed model based on bidirectional LSTM for learning embeddings of words and senses in the same semantic space (Iacobacci and Navigli, 2019). The model starts from a sense-tagged text, which is processed with a bidirectional LSTM analyzing both the preceding and the posterior context of a token s_i, where s_i is either a word or a sense tag. The output computed by the LSTM on both directions is concatenated and linearly weighted with a dense layer. Subsequently the model compares the output with the pre-trained embedding vector of the target token s_i. The training phase maximizes the similarity among the output of the network and the pre-trained embeddings: the loss is computed in terms of cosine distance.[3] LSTMEMBED pre-trained embeddings contain about 2 millions vectors. The obtained resource is featured by three sorts of representation: the word-sense representation —e.g., the vector for the sense bn:00008363n, which refers to Bank, intended as *"Sloping land, especially the slope beside a body of water"*—; the representation for a given lexicalization associated to a given sense —e.g., for the pair Bank-bn:00008363n—; and the word embedding —e.g., the vector for the term Bank—, possibly conflating all senses underlying the given term.

We finally mention LESSLEX, a multilingual lexical resource whose embeddings are grounded on the sense inventory from BabelNet and employing the word embeddings from CN Numberbatch to build a shared space for term- and sense-vectors (Colla, Mensa, and Radicioni, 2020). Each term is provided with a 'blended' terminological vector along with those describing all senses associated to that term. LESSLEX has been tested on three tasks relevant to lexical semantics: conceptual similarity, contextual similarity, and semantic text similarity, improving on or closely approaching state-of-the-art results.

Contextualized Sense Embeddings

The contextualized sense representations line of research follows directly from the introduction of contextualized language models, and strongly relies on such representations. Despite such language models provide context sensitive representations, they still lack semantic grounding to sense inventories.

The first attempt at demonstrating that contextual embeddings from pre-trained language models can be enriched by exploiting sense inventories is LMMS (Loureiro and Jorge, 2019). LMMS is an approach for generating sense embeddings relying on pre-trained contextualized language models that covers the entire WordNet 3.0 sense inventory. The proposed approach computes a list of sense embeddings starting from annotations, i.e. a sense tagged corpus. In particular, the sense vector is computed as the average of all the contextual representation for words tagged

[3]In order to generate the LSTMEMBED pre-trainied embeddings the authors chose the BabelNet 4.0 sense inventory. The BabelWiki corpus (Scozzafava et al., 2015) has been employed for both the training of the model and the representation of the objective embeddings; the latter case has been addressed using the Word2Vec's SkipGram model.

with the word sense: given n contextual embeddings c_i for a word sense s, the vector v_s is computed as $v_s = \frac{1}{n}\sum_{i=1}^{n} c_i$. Since the sense tagged corpus covers only a small percentage of the WordNet vocabulary, the authors improve sense inventory coverage exploiting WordNet structure: in order to build embeddings for higher-level abstractions, the average of the embeddings of all lower-level constituents is employed. That is, the embedding of an unseen word sense corresponds to the average of all its children representation. LMMS pre-trained embeddings cover the entire WordNet vocabulary, thus containing embeddings for $117,659$ synsets corresponding to $206,949$ unique senses. Since LMMS is grounded to the WordNet sense inventory, this resource represents vectors for English words only.

Following LMMS, SENSEMBERT has been introduced relying on the pre-trained version of BERT large (Scarlini, Pasini, and Navigli, 2020a). SENSEMBERT is a knowledge-based approach to produce latent semantic representations of word meanings in multiple languages. SENSEMBERT relies on Babelnet, Wikipedia and NASARI sense embeddings together with the pre-trained BERT large model. The proposed approach starts by collecting from Wikipedia all the sentences that are suitable for characterizing a given word synset: this is done by exploiting the link between BabelNet and Wikipedia. Once contextual information has been collected, the authors compute the contextualized word embedding for each word relevant to the target synset: relevant words for each synset are identified exploiting NASARI lexical vectors, then contextualized representation for such words are obtained through BERT large language model. Eventually, the synset embedding is built by exploiting word representations together with their rank in the NASARI lexical vector. In the same spirit of LMMS, synset representation quality is improved by exploiting the semantic network structure. Since the linking between BabelNet and Wikipedia involves nouns only, the proposed approach build representations for nouns only. Thanks to the multilingual nature of BabelNet, the authors exploited also the multilingual version of BERT to build sense embeddings for multiple languages. SENSEMBERT pre-trained embeddings contain vectors for $146,313$ senses.

ARES, so dubbed after context-AwaRe Embeddings of Senses, has been introduced few months later, as the extension of SENSEMBERT (Scarlini, Pasini, and Navigli, 2020b). ARES is a semi-supervised approach to produce sense embeddings for the lexical meanings within a lexical knowledge base; these lie in a space that is comparable to that of contextualized word vectors. The construction of ARES relies on several resources: WordNet, SyntagNet (Maru et al., 2019), UKB (Agirre, Lacalle, and Soroa, 2014) and BERT. The proposed approach starts collecting contexts for WordNet's synsets exploiting BERT: given a sense s and one of its lexicalizations l, the authors collected all occurrences of l in a corpus and computed their contextualized representation and clustered through the *k-means* algorithm. The UKB algorithm is then exploited to label each cluster with one of the senses for l. Each such cluster is then refined by exploiting the collocations from SyntagNet. Contextual information retrieved is then exploited so to build embeddings for WordNet's synsets as a combination of embeddings computed though BERT for sentences and collocations. ARES pre-trained embeddings contain vectors for 206,950 senses, covering 65% of WordNet's vocabulary (77,195 out of 117,659).

LMMS Reloaded (LMMS-R) is the most recent resource belonging to the contextualized sense embeddings family, it has been introduced as extension of LMMS (Loureiro, Jorge, and Camacho-Collados, 2022). LMMS-R is a principled approach for sense representation based on contextual language models trained exclusively with self-supervision. Following LMMS the synset embbedding for s is built by averaging the contextualized representations for the lexicalization l in a sense-tagged corpus. Such vectors are then refined exploiting the WordNet structure. LMMS-R allows for a different characterization of multiple layers according to the task for which the embeddings are designed. LMMS-R pre-trained embeddings for Word Sense Disambiguation (WSD) cover the entire WordNet vocabulary, thus containing embeddings for 117,659 synsets corresponding to 206,949 unique senses. Since LMMS-R is grounded to the WordNet sense inventory, the resource represents vectors for English words only.

5.2 Building SE-MACAROON

The algorithm for the generation of SE-MACAROON is based on an intuitive idea: collecting and storing *all* the contextual representation of word senses in a sense tagged corpus, to build *repositories* of context sensitive conceptual representations.

The idea underlying such constructive rationale is that we can better account for the precision of contextual representation computed by language models by maintaining their independence (their *contextual* profile, and thus their representational precision) rather than mixing them into a single fixed representation. Our intuition is that the embedding of a word sense s expressed with the word w may be closer to the vector for the same word sense s lexicalized with w in a similar context rather than the average of all the lexicalizations of s. In other words, our strategy relies on the hypothesis that the target word may be compared with a variety of other candidates embeddings, amongst which we can find a similar one, instead of averaging all the information into one representation. In these respects, we started from BERT contextual embeddings and we built new sense embeddings relying on WordNet and indexed on that sense inventory. The so created resource results in a collection of sets of embeddings, with each set corresponding to a particular sense (identified by a specific WordNet synset ID). Each embedding within these sets represents an instance of a word, used in a context, and exhibiting that specific sense. In Figure 5.1 we provide an example of the whole process of creation of the set of embeddings for the term *affect*, as expression of the word sense wn:00137313v — defined as *have an effect upon;*— in WordNet. We choose BERT as our starting point for several reasons: it was, at that point, one of the most popular contextual language models, with limited requirements with respect to newer language models, and it allowed to deal with higher amount of training instances in reasonable time without any loss of information (Sanh et al., 2019). After having explored BERT, we decided to produce our embeddings also using LLAMA-2, which is an open source Large Language Model, to test this approach also with a newer and bigger model. We acknowledge that the decision to use LLAMA-2, a generative language model, for extracting embeddings –a task traditionally performed by encoder architectures– may be seen as controversial. The issue of the quality of embeddings produced by generative models is not thoroughly explored in the literature and typically, *ad-hoc* models are developed specifically for generating embeddings;[4] however, these models are generally trained to produce representations of sentences or even entire documents and this approach does not align with our use case, which necessitates the

[4]OpenAI Embeddings Model - `https://platform.openai.com/docs/guides/embeddings`

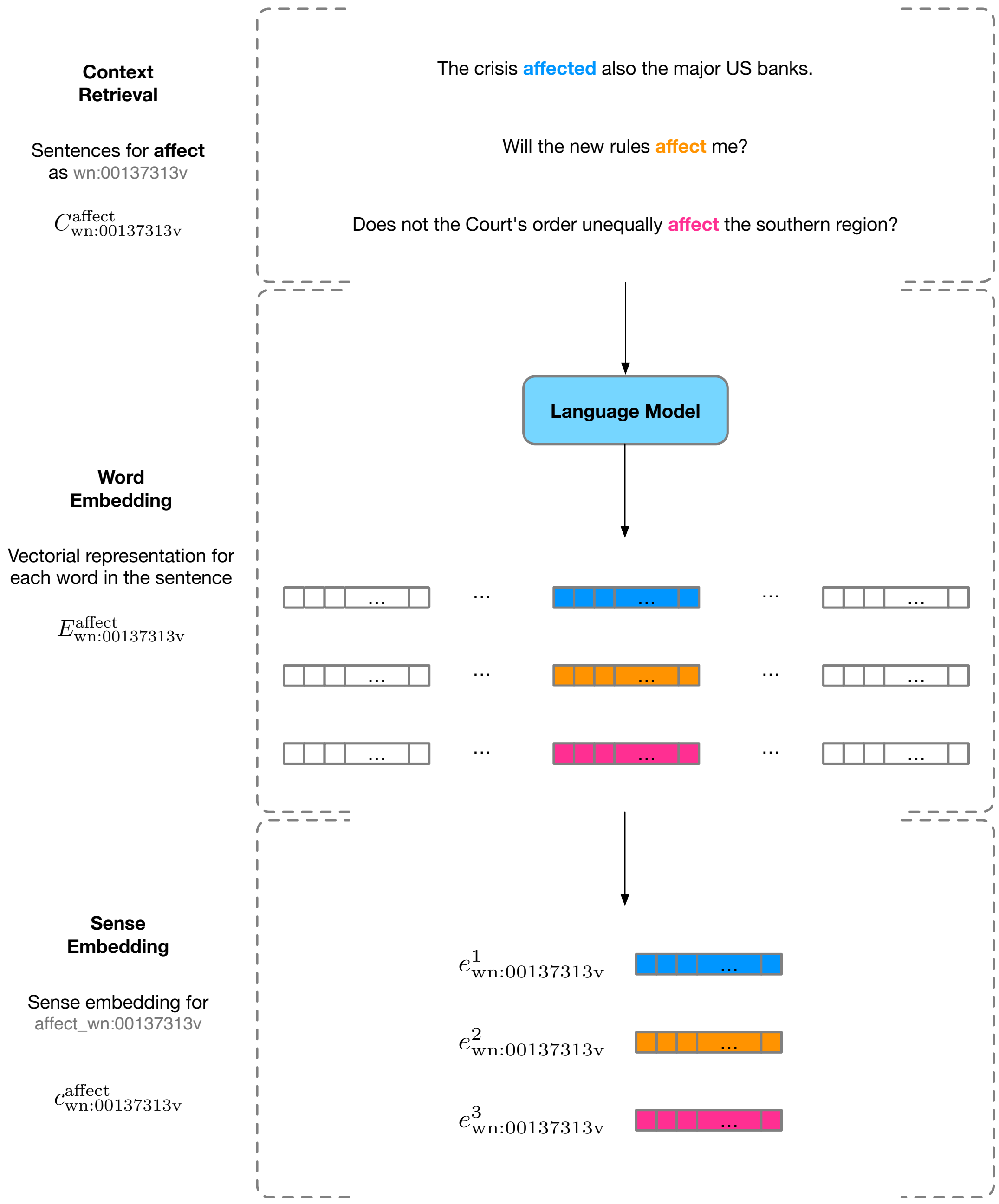

Context Retrieval
Sentences for affect as wn:00137313v
$C^{\text{affect}}_{\text{wn:00137313v}}$
The crisis affected also the major US banks.
Will the new rules affect me?
Does not the Court's order unequally affect the southern region?
Language Model
Word Embedding
Vectorial representation for each word in the sentence
$E^{\text{affect}}_{\text{wn:00137313v}}$
Sense Embedding
Sense embedding for affect_wn:00137313v
$c^{\text{affect}}_{\text{wn:00137313v}}$
$e^{1}_{\text{wn:00137313v}}$
$e^{2}_{\text{wn:00137313v}}$
$e^{3}_{\text{wn:00137313v}}$

FIGURE 5.1: Graphical illustration of the working rationale of our approach. Let us consider the word *affect* as expression of the word sense wn:00137313v. We first retrieve all the sentences in which *affect* occurs as wn:00137313v, then process them with BERT and collect the contextual representation for *affect* in our SE-MACAROON.

extraction of single word embeddings. Therefore, we opted for LLAMA-2, allowing us to straightforward access the model's last hidden layer to extract the needed embeddings. Anyway, the algorithm used with both BERT and LLAMA-2 language models is exactly the same. Our approach, described in Figure 5.1, can be divided into the following three steps:

- **Context retrieval**, to collect all the relevant sentences from a sense-tagged corpus.
- **Word embedding**, to compute the vectorial representation for each term in the sentences, given the context retrieved in the previous step.
- **Sense embedding**, to collect all the vector representations for the relevant words.

5.2.1 Context Retrieval

Each concept in SE-MACAROON is represented by a collection of vectors generated by processing sentences with the chosen language model. We start from a sense tagged corpus SC, where each instance is represented as a pair $\langle W_i, S_i \rangle$, where W_i is the sentence and S_i are the related senses. In particular, the words in the sentence $W_i = w_1^i w_2^i \dots w_k^i$ are associated with senses $S_i = s_1^i s_2^i \dots s_k^i$, where each s_j^i represents the sense with which the word w_j^i occurs in the sentence W_i. It is worth noting that, given the adopted sense inventory, not all words are actually equipped with a sense, that is, s_j^i may also be set to the null element. We adopted the sense inventory from WordNet 3.0. For example, given the sentence:

The crisis affected also the major US banks.

the related senses in the WordNet sense inventory are:

$$\varnothing \; wn{:}13933560n \; wn{:}00137313v \; \varnothing \; \varnothing \; wn{:}01472628a \; wn{:}09044862n \; wn{:}08420278n$$

where each sense corresponds to each word of the above sentence. [5]

Given the word sense s and one of its lexicalizations w, we collect all the sentences from a sense tagged corpus SC, in which w appears intended as s: in other words, we retrieve all the sentences where the sense s is lexicalized through the term w. More formally, given the pair $\langle w, s \rangle$, we define the set C_s^w containing all the pairs from SC so that:

$$C_s^w = \bigcup_{\langle w,s \rangle} \{\langle W_i, S_i \rangle\} \qquad \text{with } w_j^i \in W_i \mid w_j^i = w \land s_j^i \in S_i \mid s_j^i = s.$$

[5]Senses are represented as WordNet synset offsets, while the null element is represented with the symbol $\varnothing$.

In particular, considering again the example reported in Figure 5.1, we define $C^{\text{affect}}_{\text{wn:00137313v}}$ as the set of all sentences from SC in which the term *affect* occurs with the sense wn:00137313v.

5.2.2 Word Embedding

The second step is aimed at computing, by means of the chosen language model (as mentioned, both BERT and LLAMA-2 were explored) the representation of words in the sentences from C^w_s. First, we pre-process sentences from C^w_s with the tokenizer; every language model is equipped with its own tokenizer, like WordPiece (Wu et al., 2016) or SentencePiece (Kudo and Richardson, 2018), sentence words might have been divided into sub-words, such as, for example, the word *unequally* is decomposed to the following tokens: ⟨ *une* ⟩, ⟨ *##qual* ⟩, ⟨ *##ly* ⟩, this means that both tokens *##qual* and *##ly* are a sub-words following their preceding token. Additionally, since words might be divided in sub-tokens by the tokenizer, together with the maximum length for sentences, it may happen, depending on the considered language model, that some sentences exceed the tokens number limit, thus resulting in an inconsistent contextual representation. In this respect, it is necessary to check that the last token is consistent with the last word of the original sentence to ensure full detailed contextual representation for each such word. For example, the tokenization for the sentence could be:

Does not the Court's order unequally affect the southern region?

[⟨Does⟩, ⟨not⟩, ⟨they⟩, ⟨Court⟩, ⟨'⟩, ⟨s⟩, ⟨une⟩, ⟨##qual⟩, ⟨##ly⟩, ⟨affect⟩, ⟨the⟩, ⟨southern⟩, ⟨region⟩, ⟨?⟩]

Once sentences have been tokenized we then process the obtained representation with either the BERT or LLAMA-2 model, thus obtaining a contextual vectorial representation for each such token. The contextual embedding of an input word is computed as the average of its sub-token embeddings, that is, we define the vector for *unequally* as the average of the vector for *une*, *##qual* and of the vector for *##ly*. Regards as BERT, since the model is made of stacked layers, the typical word representation for semantic tasks is computed by summing the last four hidden layer embeddings: given that, our word embeddings are computed then as the sum of the last four hidden layer vectors (Jawahar, Sagot, and Seddah, 2019; Tenney, Das, and Pavlick, 2019; Tenney et al., 2019). Regards as the LLAMA-2 representation, the output of the last dense layer before the softmax layer is extracted. Finally, given the sentences in C_s^w, we define E_s^w as the set of contextual embedding for each word from each sentence in C_s^w; that is, E_s^w is defined as follows:

$$E_s^w = \text{LM}(W_i) \quad \forall W_i \in C_s^w, \tag{5.1}$$

where $\text{LM}(W_i)$ is the embedding function that produces contextual representation e_j^i for each word in the sentence. Therefore, for the sentence $W_i = w_1^i, \dots w_k^i$, the function $\text{LM}(W_i)$ is defined as $\text{LM}(W_i) = e_1^i e_2^i \dots e_k^i$ where e_j^i is the contextual representation for the j-th token in the i-th sentence W_i.

5.2.3 Sense Embedding

In this final step, we build a representation for each target sense in the sense tagged corpus; starting from the set of embeddings E_s^w we retrieve all the representations for the sense s lexicalized with the word w, and collect them as our representation for the word sense. In particular, we define c_s^w as the collection of all the embeddings e_j^i from E_s^w that encode ($\dashleftarrow$ in the Equation 5.2) the word w conveying the sense s:

$$c_s^w = \bigcup_{W_i, S_i} \{e_j^i \dashleftarrow \langle w_j^i, s_j^i \rangle\} \qquad \text{with } \; w_j^i = w \wedge s_j^i = s; \;\; w_j^i \in W_i, \; s_j^i \in S_i. \tag{5.2}$$

Here w_j^i and s_j^i are the word and its related sense in the i-th dataset instance $\langle W_i, S_i \rangle$. Let us consider the example in Figure 5.1: the representation for $c_{\text{wn:00137313v}}^{\text{affect}}$ is defined as the collection of the three word embeddings for *affect* resulting from the application of either model to the three sentences in which *affect* is intended as conveying the word sense wn:00137313v. At the end of the three steps each sense in the sense tagged corpus SC is provided with a set of associated vectorial representations, made of the collection of lexicalizations of the given sense.

It is worth noting that the sense embeddings c_s^w included in SE-MACAROON are indexed on both the sense s and the lexicalization w. That is, the sense embeddings

TABLE 5.1: Figures on the generation process of SE-MACAROON, divided by Part of Speech. The average occurrences per sense are reported together with their standard deviation (σ).

SE-MACAROON Statistics	All	Nouns	Verbs	Adjs	Advs
Sense (term, synset) Vectors	31,352	14,893	8787	5939	1733
Senses Occurrences	199,363	76,062	78,820	27,787	16,694
AVG Occurrences per Sense	6.36 ($\pm$66.72)	5.11 ($\pm$50.48)	8.97 ($\pm$105.00)	4.68 ($\pm$14.79)	9.63 ($\pm$43.83)
Occurrences per Sense Range	$[1, 9013]$	$[1, 5846]$	$[1, 9013]$	$[1, 413]$	$[1, 1475]$
Sense (synset) Vectors	24,528	12,418	5794	5016	1320

c_s^w representing the exact expression of the word sense s lexicalized with the word w are actually the collection of all the occurrences of s expressed by means of w.

5.3 Evaluation

In this section we report the experimental settings in which we conducted the evaluation of SE-MACAROON when testing on English Word Sense Disambiguation task. In what follows we introduce the training set along with some figures describing the resource; we then introduce the test sets and the system setup. We finally report the results along with the discussion.

5.3.1 SE-MACAROON training dataset

SemCor We trained[6] SE-MACAROON sense embeddings on SemCor (Miller et al., 1994); SemCor is a manually sense-tagged corpus, divided in 352 documents for a total of 226,040 sense annotations; it is, to the best of our knowledge, the largest corpus manually annotated with WordNet senses. We started from a total of 37,176 sentences contained in SemCor; we then retained 33,399 after the filtering step, thus resulting in 199,363 sense annotations.[7] The final figures of the resource and details concerning its generation are reported in Table 5.1. The final number of couples *word_sense* in SE-MACAROON, using only SemCor as training set, amounts to 31,352, corresponding to 24,548 unique synsets, covering only about the 21% of the total 117,659 WordNet synsets. The number of occurrences per sense ranges from 1

[6]By "trained" we mean "produced with the chosen language model all the necessary embeddings"
[7]Words tagged with a Wordnet SynsetID

to 9013 for the verb *to be*. Unfortunately, an important factor to consider along with numbers coming from the training dataset, is the coverage that the dataset ensures on the test dataset: As detailed reported in Table 5.8, the coverage from SemCor was lacking of 707 couples lemma-sense, and was not enough to compete with other state of the art models. The solution we adopted is described in the following paragraph.

SemCor + Gemini Coverage issues were dealt with by exploiting Gemini (Anil et al., 2023). The idea behind this data augmentation step is simple: some of the pairs sense-lemma needed in the test set are not actually present in the training corpus. Gemini offers us the possibility to generate sentences containing those lemmas, hopefully used in the desired sense. In particular we exploited prompt engineering in order to make Gemini produce sentences in as controlled as possible fashion, by using the following prompt:

Produce a sentence with the ⟨desired Part of Speech⟩ ⟨desired lemma⟩ used with the following definition: ⟨wordnet sense definition⟩.

For example, considering the lemma *bank*, used as *noun*, in the sense of *sloping land (especially the slope beside a body of water)*, as reported in Wordnet sysnet, the prompt would have looked like this:

Produce a sentence with the noun bank used with the following definition: sloping land (especially the slope beside a body of water).

After this first set of production, we decided to enhance the possibility of producing sentences containing the lemma used in the right sense by adding to the prompt a series of keywords Gemini to be employed in the produced sentence. Such keywords were added by associating to every pair lemma-sense the words extracted using KeyBert (Grootendorst, 2020) from SemCor and ALL corpora sentences, were the specific lemma-sense pair occurred. The second prompt looked like:

Produce a sentence with the ⟨desired Part of Speech⟩ ⟨desired lemma⟩ used with the following definition: ⟨wordnet sense definition⟩. The sentence should contain the following keywords: ⟨keywords list⟩

The final set of generated sentences, which we used for the experiments, includes both sentences generated with such keywords and sentences without them. Gemini output was finally filtered in order to verify the satisfaction of the main structural elements of the request: for instance, the presence of the lemma and the part of speech in which the lemma was used. Sentences containing specific utterances such as *I'm sorry, I cannot satisfy your request*, were filtered, since these indicate that Gemini was not able to generate the expected sentence. The structural analysis cannot guarantee the correct usage of the lemma in the desired sense, but that would require to have the WSD problem already solved.

To complete this data-centered paragraph, we just want to describe the evaluation devised for this data augmentation step. This sort of assessment is kept separated from the extrinsic evaluation of the additions by directly using them in SE-MACAROON, whose results are described in the 5.3.5 Section. This *intrinsic* evaluation consisted of manually assessing the usage of a certain lemma-sense pair within a sentence, for 100 randomly chosen senses out of the 707 ones needed to obtain complete coverage on our test-set. This resulted in a 90% of sentences tagged as "containing the couple lemma-sense used in the required sense". The descriptive statistics associated to the additions obtained through Gemini are presented in Table 5.2: the final number of sense embeddings after Gemini data augmentation amounts to $32,059$, exactly 707 pairs lemma-sense more than the corpus only including SemCor. These pairs overall correspond to $25,064$ unique synsets in Wordnet. About the distribution over part-of-speech tags, the augmentation was stronger for nouns, where 443 pairs lemma-sense were added, less relevant for verbs (139 additions) and adjectives (102 additions), and minimal for adverbs where only 23 pairs where added. On average addition for every pair 44.72 example sentences were added, with standard deviation of 6.00.

5.3.2 Evaluation Benchmarks

We assessed SE-MACAROON vectors on the Word Sense Disambiguation task as it constitutes the most popular and obvious task for evaluating sense embeddings (Loureiro, Jorge, and Camacho-Collados, 2022). WSD is a long-standing challange in the Natural Language Processing, as it lies at the core of language understanding (Navigli, 2009): it is defined as the task of associating words in context with their most suitable meaning from a pre-defined sense inventory. Given the target word w_t and

TABLE 5.2: Figures on the generation process of SE-MACAROON after the data augmentation process with Gemini, divided by Part of Speech. The average occurrences per sense are reported together with their standard deviation (σ).

SE-MACAROON Statistics	All	Nouns	Verbs	Adjs	Advs
Sense (term, synset) Vectors	32,059	15,336	8926	6041	1756
Senses Occurrences	229,953	95,159	84,970	32,105	17,719
AVG Occurrences per Sense	7.17 ($\pm$66.22)	6.20 ($\pm$55.18)	9.52 ($\pm$104.27)	5.31 ($\pm$15.51)	10.09 ($\pm$43.73)
Occurrences per Sense Range	[1, 9013]	[1, 5846]	[1, 9013]	[1, 413]	[1, 1475]
Sense (synset) Vectors	25,064	12,764	5884	5083	1333

the sentence $W = \{w_1 \dots w_t \dots w_l\}$, the task consists in associating w_t in W with a sense taken from a given sense inventory. For example, given the word *bark* and the sentence *The tree's bark was dark.* Let us consider WordNet as our reference sense inventory: in such example the task is to provide the correct entry form the WordNet's senses for *bark*, that is wn:13162297n defined as *tough protective covering of the woody stems and roots of trees and other woody plants*.

We carried out the evaluation on the test sets in the English WSD framework (Raganato, Camacho-Collados, and Navigli, 2017). The benchmark includes five standardized evaluation datasets from the past Senseval-SemEval competitions, that are:

- Senseval-2 (SE02), consisting of 2283 sense annotations (Edmonds and Cotton, 2001);
- Senseval-3 (SE03), consisting of 1850 sense annotations (Snyder and Palmer, 2004);
- SemEval-2007 (SE07), consisting of 455 sense annotations (Pradhan et al., 2007);
- SemEval-2013 (SE13), consisting of 1644 sense annotations (Navigli, Jurgens, and Vannella, 2013); and
- SemEval-2015 (SE15), consisting of 1022 sense annotations (Moro and Navigli, 2015).

The whole benchmark includes the concatenation of the five test sets, and in the following it is referred to as the ALL dataset.

5.3.3 System Setup

The SE-MACAROON word sense disambiguation strategy can be arranged into the following three steps: *(i)* sentence embedding, aimed at computing contextual representations for the input sentence, *(ii)* sense occurrences scoring, aimed at computing a score for each sense's occurrence, and *(iii)* word sense ranking, where sense occurrences are ranked based on their score and the word sense is selected through a majority voting among the top N entries. The working rationale of the WSD strategy is presented in Figure 5.2.

Sentence Embedding The first stage of the computation is targeted to compute the contextual representation for the input sentence. Therefore, given the sentence $W = \{w_0, w_1, \dots, w_t, \dots, w_k\}$, containing the target word w_t, we process the tokenized sentence with either BERT or LLAMA-2, thereby obtaining contextual embeddings $E = \{e_1, \dots, e_t, \dots, e_k\}$ for each token in W. Once again, the sub-token vectors are averaged to compute a token-level representation. Additionally, similar to the building approach, we define the embedding e_i for a word w_i as the sum of the last four layers of the employed language model. Since we are interested in assessing the contribution of the other words in the sentence, particularly those surrounding the target word w_t, we retain the contextual representation for those words occurring within a given span d from w_t only: a context window is defined, surrounding the target word w_t, $CTX_{w_t} = \{e_{t-d}, \dots, e_{t-1}, e_{t+1}, \dots, e_{t+d}\}$ including d word representations preceding and following w_t. Let us consider the Figure 5.2, the input sentence is *We must believe we have the ability to affect our own destinies: otherwise why try anything?* where *affect* is the target term. In the example our context window size is set to 3, which means that we retain the contextual embeddings for the three words preceding w_t and for the three words following w_t: the left context is *the ability to* and the right context is *our own destiny*, such that $CTX_{\text{affect}} = [e_6, e_7, e_8, e_{10}, e_{11}, e_{12}]$.

Senses occurrences scoring After having built the contextual representation for the target word along with its context, we compute a score for each occurrence $e_s^i \in c_s^{w_t}$ of word senses for our target word w_t to retrieve the most likely sense. In order to disambiguate w_t, we start by retrieving all the senses for the target

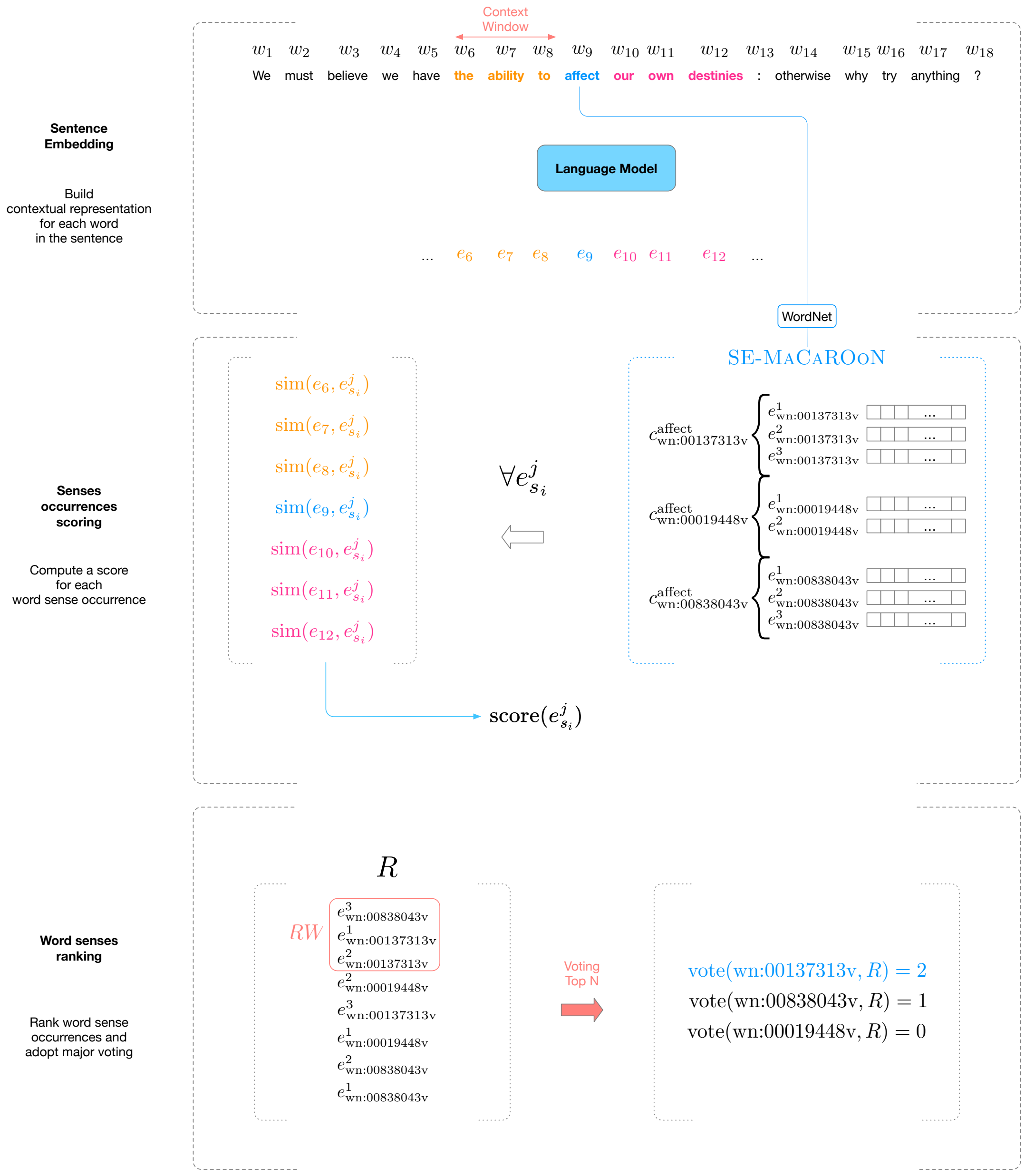

Context Window
w_1 w_2 w_3 w_4 w_5 w_6 w_7 w_8 w_9 w_{10} w_{11} w_{12} w_{13} w_{14} w_{15} w_{16} w_{17} w_{18}
We must believe we have the ability to affect our own destinies : otherwise why try anything ?
Sentence Embedding
Build contextual representation for each word in the sentence
Language Model
... e_6 e_7 e_8 e_9 e_{10} e_{11} e_{12} ...
WordNet
SE-MaCaROoN
Senses occurrences scoring
Compute a score for each word sense occurrence
$\mathrm{sim}(e_6, e^j_{s_i})$
$\mathrm{sim}(e_7, e^j_{s_i})$
$\mathrm{sim}(e_8, e^j_{s_i})$
$\mathrm{sim}(e_9, e^j_{s_i})$
$\mathrm{sim}(e_{10}, e^j_{s_i})$
$\mathrm{sim}(e_{11}, e^j_{s_i})$
$\mathrm{sim}(e_{12}, e^j_{s_i})$
$\forall e^j_{s_i}$
$c^{\text{affect}}_{\text{wn:00137313v}}$
$e^1_{\text{wn:00137313v}}$ ...
$e^2_{\text{wn:00137313v}}$ ...
$e^3_{\text{wn:00137313v}}$ ...
$c^{\text{affect}}_{\text{wn:00019448v}}$
$e^1_{\text{wn:00019448v}}$ ...
$e^2_{\text{wn:00019448v}}$ ...
$c^{\text{affect}}_{\text{wn:00838043v}}$
$e^1_{\text{wn:00838043v}}$ ...
$e^2_{\text{wn:00838043v}}$ ...
$e^3_{\text{wn:00838043v}}$ ...
$\mathrm{score}(e^j_{s_i})$
Word senses ranking
Rank word sense occurrences and adopt major voting
R
RW
$e^3_{\text{wn:00838043v}}$
$e^1_{\text{wn:00137313v}}$
$e^2_{\text{wn:00137313v}}$
$e^2_{\text{wn:00019448v}}$
$e^3_{\text{wn:00137313v}}$
$e^1_{\text{wn:00019448v}}$
$e^2_{\text{wn:00838043v}}$
$e^1_{\text{wn:00838043v}}$
Voting Top N
$\mathrm{vote}(\text{wn:00137313v}, R) = 2$
$\mathrm{vote}(\text{wn:00838043v}, R) = 1$
$\mathrm{vote}(\text{wn:00019448v}, R) = 0$

FIGURE 5.2: Graphical illustration of the working rationale of the WSD strategy. Let us consider the word *affect*, as our target word, occurring in the sentence *We must believe that we have the ability to affect our own destinies: otherwise why try anything?*. At first we build contextual embeddings for each word in the sentence through BERT; then we retain only the representations for $W (W = 3)$ words from the left (left context, orange in the figure) and W words from the right (right context, pink in the figure) of the target word, along with the embedding for *affect*. We then compute the similarity between the context, including the target word, and every occurrence $e^j_{s_i}$ of each sense of *affect* in SE-MaCaROoN; we then define the score for e^i_s as the weighted average of the similarities. Eventually, we rank all the occurrences $e^j_{s_i}$ of each word sense for *affect* and extract, through majority voting, the sense wn:00137313v as the most likely sense considering the top N ($N = 3$) occurrences of the ranking.

word in the WordNet sense inventory, thus obtaining the set of candidate senses $S^{w_t} = \{s_1^{w_t}, s_2^{w_t}, \ldots, s_n^{w_t}\}$ as all the word senses for w_t for which a correspondence can be found in SE-MACAROON. Since we refer to senses for w_t, in what follows the superscript w_t is dropped to simplify the notation: s_i will thus be intended as $s_i^{w_t}$. For the sake of the readability, we recall here that each SE-MACAROON entry $c_{s_i}^{w_t}$ is defined as $c_{s_i}^{w_t} = [e_{s_i}^1, e_{s_i}^2, \dots, e_{s_i}^n]$, that is, the set of occurrences of the word senses s_i expressed by the word w_t, thus we retain only WordNet's senses $s_i^{w_t}$ with a corresponding SE-MACAROON entry $c_{s_i}^{w_t}$. The scoring step is aimed at computing a score for each such sense occurrence $e_{s_i}^j$, expressing the semantic similarity between $e_{s_i}^j$ and w_t along with its related context CTX_{w_t}. Therefore, given a candidate word sense s_i and its occurrence $e_{s_i}^j$ for the target word w_t, we defined different scoring functions.

α score function The first scoring function is defined as follows:

$$\text{score}(e_{s_i}^j) = \left(\frac{1}{2d+1}[\alpha \cdot \text{sim}(e_t, e_{s_i}^j)\right) + \sum_{k \in [t-d,\dots,t-1,t+1,\dots,t+d]} \left(\frac{(1-\alpha)}{2d} \cdot \text{sim}\left(e_k, e_{s_i}^j\right)\right), \tag{5.3}$$

where e_t and e_k indicate a given embedding representation for the target word and the words in the context CTX_{w_t} respectively, while sim is the cosine similarity function between the two vectors at hand. The α parameter is used to tune up the balance between the relevance of the similarity among the word sense and the target word representation, and the similarity between the word sense and the context words representations. α default value was set to 0.5 (it's value is further analyzed in Section 5.3.5). We end up with a score for each occurrence $s_{s_i}^j$ of the candidate word sense s_i from S^{w_t}.

Weighted score For the second scoring function we got rid of the α parameter, and instead the context is directly used as weight for the similarity between e_t and e_k.

$$\text{score}(e_{s_i}^j) = \text{sim}(e_t, e_{s_i}^j) \cdot \frac{1}{|K|} \sum_{k \in K = [t-d,\dots,t-1,t+1,\dots,t+d]} \text{sim}(e_k, e_{s_i}^j), \tag{5.4}$$

In this specific case we opted for using the mean value of the similarities of the elements of the context as weight for the similarity between e_t and e_k, while in a variant of the formula we also explored the max value:

$$\text{score}(e^j_{s_i}) = \text{sim}(e_t, e^j_{s_i}) \cdot \max_{k \in K=[t-d,\dots t-1,t+1,\dots t+d]} (\text{sim}(e_k, e^j_{s_i})), \tag{5.5}$$

Let us consider again the Figure 5.2, showing that SE-MACAROON contains three different sense representations for *affect*: $c^{\text{affect}}_{\text{wn:00137313v}}$, $c^{\text{affect}}_{\text{wn:00019448v}}$ and $c^{\text{affect}}_{\text{wn:00838043v}}$. Since each SE-MACAROON entry is provided with multiple occurrences, we compute the similarity between the contextual representation from e_6 to e_{12} and each word sense occurrence $e^j_{s_i}$.[8] We therefore compute a score for $e^j_{s_i}$ according to one of the equations 5.3, 5.4 or 5.5: in the first case the similarity between e_9 and $e^j_{s_i}$ is weighted by α, while each similarity score defined between the context words and $e^j_{s_i}$ is equally weighted by $\frac{(1-\alpha)}{6}$, where $d = 3$, thus obtaining $2d = 6$.

Word senses ranking Once we have computed a score for each occurrence for senses in S^{w_t} we proceed with the ranking step. We therefore define the ranking R as the list of occurrences of senses $e^j_{s_i}$ sorted in descending order, based on their associated score. In order to retrieve the most likely word sense for w_t we adopt a majority voting strategy on the top N items of the ranking R. More precisely, we *i*) define a window RW containing the top N ranked items as $RW = [e^j_{s_i}, \dots, e^l_{s_k}]$; *ii*) we count the number of times in which a sense occurs in RW; and *iii*) we select the top scoring sense. More formally, for each word sense s_i in S^{w_t}, we define the following voting function:

$$\text{vote}(s_i, R) = \sum_{e^j_{s_k} \in RW} \text{count}(s_i, e^j_{s_k}), \tag{5.6}$$

where the count function returns 1 if $s_i = s_k$, that is:

$$\text{count}(s_i, e^j_{s_k}) = \begin{cases} 1 & \text{if } s_i = s_k \\ 0 & \text{otherwise} \end{cases}. \tag{5.7}$$

Once all the senses from S^{w_t} have been provided with a score through the voting function, we select the most voted word sense for our target word w_t. More formally,

[8] e_9 is the contextualized embedding for the target word *affect*, while $[e_6, e_7, e_8, e_{10}, e_{11}, e_{12}]$ represent the context CTX_{affect}.

to retrieve the most voted word sense, we select s_* such that

$$s_* = \arg\max_{s_i \in S^{w_t}} \text{vote}(s_i, R), \tag{5.8}$$

that is, s_* is the most voted word sense in the top N elements of the ranking R. Let us again consider the example reported in Figure 5.2: here the ranking R is built by sorting the occurrences for the three senses wn:00137313v, wn:00838043v and wn:00019448v for *affect*. In the example, the size N of window RW is set to 3, thus obtaining two votes for the sense wn:00137313v and one vote for wn:00838043v. Therefore, we select the sense wn:00137313v for the target word *affect* in our input sentence. Should multiple senses receive the same number of votes in then ranking step, the choice is made based on the similarity value of the candidates of those senses: we select the sense whose candidate obtained the max similarity value.

5.3.4 Evaluation Metrics

Precision (P), Recall (R), and their harmonic mean (F1) metrics have been largely accepted from the literature to assess the performance of systems on the Word Sense Disambiguation task. The mentioned metrics have been adopted to overcome the Accuracy drawback: the accuracy focuses on the positive class, giving no intuition on the system's performances on the negative class (Edmonds and Cotton, 2001; Cohn, 2003). Following the literature, we define the precision as the fraction of correctly predicted senses within the set of instances for which the algorithm provided an answer, while the recall is computed as the proportion of correctly predicted senses out of the set of correct answers. More formally, the employed metrics are defined as follows:

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{|D|} \qquad F1 = 2 * \frac{P * R}{P + R} \tag{5.9}$$

where TP (True Positives) are the correctly predicted instances; FP (False Positives) denote the instances for which the system provided a wrong prediction out of the $|D|$ tagged instances in the evaluation benchmark. It is worth noting that $TP + FP$ corresponds to the number of instances for which the system under evaluation was able

TABLE 5.3: Results on the SemEval 17 English dataset. The reported figures express the precision (P) and recall (R) metrics along with their harmonic mean F1 calculated according to the different reported equations. For all the lines in this table we adopted the same configuration: $\alpha = 0.5$ (where applicable), d=3, N=5

LM	Corpus	Scoring function	ALL n = 7253		
			P	R	F1
BERT	SemCor	Eq. 5.3	0.7548	0.6785	0.7146
BERT (AVG)	SemCor	Eq. 5.3	0.6989	0.6283	0.6617
LLama-2	SemCor	Eq. 5.3	0.7389	0.6676	0.7014
LLama-2 (AVG)	SemCor	Eq. 5.3	0.6887	0.6222	0.6537
BERT	SemCor + Gemini	Eq. 5.3	0.7591	0.7591	0.7591
BERT (AVG)	SemCor + Gemini	Eq. 5.3	0.6994	0.6994	0.6994
BERT	SemCor + Gemini	Eq. 5.4	0.7600	0.7600	0.7600
BERT	SemCor + Gemini	Eq. 5.5	0.7601	0.7601	**0.7601**
LLama-2	SemCor + Gemini	Eq. 5.3	0.7415	0.7415	0.7415
LLama-2 (AVG)	SemCor + Gemini	Eq. 5.3	0.6772	0.6772	0.6772
LLama-2	SemCor + Gemini	Eq. 5.4	0.7419	0.7419	0.7419
LLama-2	SemCor + Gemini	Eq. 5.5	0.7364	0.7364	0.7364

to provide a prediction; therefore, we define the sum of true positives and false positives as *Coverage*. We report this figure, as well, to complement the results recorded through the recall metrics: in fact, the recall scores as an error both actual errors and uncovered senses for which the system was unable to provide a prediction.

5.3.5 Results

We compared SE-MACAROON with the most recent lexical resources representing word senses as contextualized embeddings. All the assessed resources built sense embeddings with the same language model, BERT Large.[9] Given the definition for the evaluation metrics provided in Equation 5.9 and according to the building principles underlying each resource, we adopted different disambiguation strategies. Since LMMS, SENSEMBERT, SENSEMBERT$_{sup}$, and ARES representations are twice as large as the BERT representations,[10] we repeated the BERT embedding (this step was implemented based on the literature presenting each and every employed resource) of the target word to match the number of dimensions. In other words, the

[9]Available on the TensorFlow Hub repository at `https://tfhub.dev/tensorflow/bert_en_cased_L-24_H-1024_A-16/4`.

[10]BERT Large embeddings have 1024 dimensions while the resource's representations are provided with 2048 dimensions.

TABLE 5.4: Analysis over the three formulation of the best *d* parameter (context window size) configuration. In this configuration only *d* changes. The training corpus considered is the one obtaining the best results in table 5.3, SemCor+Gemini.

d value	F1 score Eq. 5.3	F1 score Eq. 5.4	F1 score Eq. 5.5
0	0.7574	0.7574	0.7574
3	0.7591	0.7600	0.7601
5	0.7580	0.7564	0.7616
10	0.7582	0.7583	0.7629
20	0.7589	0.7572	**0.7641**
30	0.7584	0.7551	0.7626
50	0.7584	0.7557	0.7614

embedding was concatenated with itself in order to yield 2048 dimensions and to be able to compare the resulting vectors with those from the mentioned resources employing 2048-sized vectors. Conversely, LMMS-R embeddings match the number of dimensions of the representations produced through BERT.

For all the assessed resources, the adopted disambiguation strategies are the 1-nearest neighbor and our strategy that takes into consideration also context. For the first one, for each target word w in the test set we computed its contextual embedding by means of BERT, and compared it against the embeddings of the assessed resource associated with the senses of w. For the latter, other resources embeddings were simply used in our code instead of SE-MACAROON. The Most Frequent Sense heuristics is customarily adopted in literature as the backoff strategy for non covered instances —i.e., predicting the most frequent sense of a lemma in WordNet for instances unseen at training time—, however, to assess the precision of each such resource we decided not to make use of the backoff strategy. In Table 5.8 we report the coverage for each resource and each benchmark in the evaluation framework.

Together with the state of the art resources, we also compared SE-MACAROON to SE-MACAROON$_{AVG}$ that has been devised by following the constructive rationale of our resource and by also averaging all sense occurrences into a single vectorial representation for each pair $\langle w_i, s_i \rangle$. In Table 5.7 we report results for both BERT and LLAMA-2 models used to build SE-MACAROON.

Parameters optimization Parameters employed by the SE-MACAROON and SE-MACAROON$_{AVG}$ systems are α (Equation 5.3); the size d of the context window for

TABLE 5.5: Results on the SemEval 17 English dataset, reported by the cited articles presenting each resource

Resource	SE02 n = 2282			SE03 n = 1850			SE07 n = 445			SE13 n = 1644			SE15 n = 1022			ALL n = 7253		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
LMMS [1]	76.3	76.3	76.3	75.6	75.6	75.6	68.1	68.1	68.1	75.1	75.1	75.1	77.0	77.0	77.0	75.4	75.4	75.4
LMMS-R [2]	76.7	76.7	76.7	74.1	74.1	74.1	66.4	66.4	66.4	75.2	75.2	75.2	77.6	77.6	77.6	75.2	75.2	75.2
ARES [3]	78.0	78.0	78.0	77.1	77.1	77.1	71.0	71.0	71.0	77.3	77.3	77.3	83.2	83.2	83.2	77.9	77.9	77.9
SENSEMBERT and SENSEMBERT$_{sup}$ tested only on nouns																		
SENSEMBERT [4]	80.6	80.6	80.6	70.3	70.3	70.3	73.6	73.6	73.6	74.8	74.8	74.8	80.2	80.2	80.2	75.9	75.9	75.9
SENSEMBERT$_{sup}$ [4]	83.7	83.7	83.7	79.7	79.7	79.7	79.9	79.9	79.9	78.7	78.7	78.7	80.2	80.2	80.2	80.4	80.4	80.4

[1] Loureiro and Jorge, 2019 (`https://github.com/danlou/LMMS/tree/LMMS_ACL19`)
[2] Loureiro, Jorge, and Camacho-Collados, 2022 (`https://github.com/danlou/LMMS`)
[3] Scarlini, Pasini, and Navigli, 2020b
[4] Scarlini, Pasini, and Navigli, 2020a

the target word CTX_{w_t}; and the size N of the ranking window RW, while the number of occurrences for each word sense was not limited.[11] While testing different scoring functions (Equations 5.4 and 5.5), no α parameter was needed.

In the discussion Section we are going to explain all the results and all the tests we made, also to elaborate on the best parameter configuration.

Discussion The results obtained in the standard test sets of the WSD Evaluation Framework by (Raganato, Camacho-Collados, and Navigli, 2017) are reported in Table 5.3, 5.4 and 5.7.

From Table 5.3, we can observe differences stemming from the adoption of different equations and parameter settings. One first comparison involves the representational precision of the two language models employed to produce the embeddings, BERT and LLAMA-2. In this case BERT, despite the reduced vector size (1024 elements against 4096 of LLama embeddings), is clearly able to create more meaningful representations, obtaining consistently better results on WSD, regardless of the considered experimental setting. This, as mentioned when introducing the usage of LLAMA-2, can be explained by taking into account the fact that LLAMA-2 was not specifically conceived to extract embeddings; reported results also show that smaller models can provide valuable results. Given this fact, by starting from Table 5.4 we report the results obtained by employing BERT to produce SE-MACAROON embeddings.

[11] In the following we report the results for N set to 5, which experimentally provided the best accuracy.

TABLE 5.6: Results on the SemEval 17 English dataset of all the mentioned resources, calculated using $d = 0$ and $\alpha = 1$, 1NN condition. The figures reported express the precision (P) and recall (R) metrics along with their harmonic mean F1 calculated according to Equation 5.9.

Resource	SE02 n = 2282			SE03 n = 1850			SE07 n = 445		
	P	R	F1	P	R	F1	P	R	F1
LMMS	0.7432	0.7432	0.7432	0.7135	0.7135	0.7135	0.6088	0.6088	0.6088
SENSEMBERT	0.8170	0.4518	0.5818	0.7664	0.4292	0.5502	0.7486	0.3011	0.4295
SENSEMBERT$_{sup}$	0.8471	0.4684	0.6033	0.8359	0.4681	0.6001	0.7541	0.3033	0.4326
LMMS-R	0.7226	0.7226	0.7226	0.6838	0.6838	0.6838	0.5758	0.5758	0.5758
ARES	0.7577	0.7577	0.7577	0.7357	0.7357	0.7357	0.6725	0.6725	**0.6725**
SE-MACAROON	0.7695	0.7695	**0.7695**	0.7497	0.7497	**0.7497**	0.6549	0.6549	0.6549

Resource	SE13 n = 1644			SE15 n = 1022			ALL n = 7253		
	P	R	F1	P	R	F1	P	R	F1
LMMS	0.7318	0.7318	0.7318	0.7593	0.7593	0.7593	0.7269	0.7269	0.7269
SENSEMBERT	0.7299	0.7299	0.7299	0.7962	0.4932	0.6091	0.7705	0.5054	0.6104
SENSEMBERT$_{sup}$	0.7695	0.7695	**0.7695**	0.8152	0.5049	0.6236	0.8100	0.5314	0.6417
LMMS-R	0.7074	0.7074	0.7074	0.7358	0.7358	0.7358	0.7019	0.7019	0.7019
ARES	0.7500	0.7500	0.7500	0.8014	0.8014	**0.8014**	0.7511	0.7511	0.7511
SE-MACAROON	0.7555	0.7555	0.7555	0.7935	0.7935	0.7935	0.7575	0.7575	**0.7575**

Secondly, the data augmentation approach exploiting the generative features provided by Gemini was helpful in granting the possibility to cover the whole set of senses contained in the test set, with a raise of the F1 score in the order of 5%. The F1 scores for SE-MACAROON are systematically higher than the F1 scores for SE-MACAROON$_{AVG}$ in a range from 1% to 4% in all considered settings, contributing to confirm the claim that maintaining embeddings independence (their contextual profile, and thus their representational precision) rather than mixing them into a single fixed representation is a good choice.

Shifting the attention to the scoring functions, (please refer to the results provided in Table 5.4), we can observe that Eqs. 5.4 and 5.5 allow obtaining improved results over Eq. 5.3, giving us the possibility to get rid of the α parameter and so to consider a simpler function for our ranking calculations. In other words, we can directly use context embeddings similarity with our candidate as weight for the similarity between the candidate sense itself and the target word, instead of using a fixed parameter to decide which part of the formula should have the higher importance. Given that the approach implementing Eq. 5.3 relies on the α parameter (balancing the similarity between the target word and the word sense occurrence $e^j_{s_k}$ and the similarity between the context and $e^j_{s_k}$), we investigated the effect of varying the balancing factor. The α parameter was varied in a range $[0, 1]$ with 0.1 step, and we

TABLE 5.7: Results on the SemEval 17 English dataset of all the mentioned resources, calculated using the best condition for our resource, using Eq. 5.5 with $d = 20$. The figures reported express the precision (P) and recall (R) metrics along with their harmonic mean F1 calculated according to Equation 5.9.

Resource	SE02 n = 2282			SE03 n = 1850			SE07 n = 445		
	P	R	F1	P	R	F1	P	R	F1
LMMS	0.6919	0.6919	0.6919	0.6335	0.6335	0.6335	0.5516	0.5516	0.5516
SENSEMBERT	0.7884	0.4360	0.5615	0.7037	0.3941	0.5052	0.7049	0.2835	0.4044
SENSEMBERT$_{sup}$	0.8074	0.4465	0.5751	0.7539	0.4222	0.5412	0.7541	0.3033	0.4326
LMMS-R	0.6543	0.6543	0.6543	0.5935	0.5935	0.5935	0.5099	0.5099	0.5099
ARES	0.7112	0.7112	0.7112	0.6741	0.6741	0.6741	0.5978	0.5978	0.5978
SE-MACAROON	0.7805	0.7805	**0.7805**	0.7530	0.7530	**0.7530**	0.6659	0.6659	**0.6659**

Resource	SE13 n = 1644			SE15 n = 1022			ALL n = 7253		
	P	R	F1	P	R	F1	P	R	F1
LMMS	0.6210	0.6210	0.6210	0.7025	0.7025	0.7025	0.6537	0.6537	0.6537
SENSEMBERT	0.6448	0.6448	0.6448	0.7457	0.4618	0.5704	0.7114	0.4667	0.5636
SENSEMBERT$_{sup}$	0.6807	0.6807	0.6807	0.7662	0.4746	0.5861	0.7444	0.4883	0.5898
LMMS-R	0.6040	0.6040	0.6040	0.6869	0.6869	0.6869	0.6229	0.6229	0.6229
ARES	0.6582	0.6582	0.6582	0.7554	0.7554	0.7554	0.6888	0.6888	0.6888
SE-MACAROON	0.7579	0.7579	**0.7579**	0.8014	0.8014	**0.8014**	0.7641	0.7641	**0.7641**

verified that the best configuration was with $\alpha = 0.5$; results reported in Table 5.3 have been obtained with this setting.

Considering then only formulations 5.4 and 5.5, we observed that using the max function instead of the mean, is beneficial in particular when using larger context around the target word. As we can see in Table 5.4, it seems that averaging similarity values is detrimental when increasing the value of the d parameter: conversely, using the max function like in Eq. 5.5 is helpful to achieve the best performing parameter setting with an F1 score of 76.41, considering a context window size of 20 tokens.

We also compared SE-MACAROON with all the other mentioned state of the art resources. To be completely fair in the presentation of the results for other resources, in Table 5.5 we also present all the results reported in literature in the works introducing all the considered resources. In this Table, LMMS, LMMS-R and ARES are tested over all instances in all datasets, while SENSEMBERT and SENSEMBERT$_{sup}$ results are not directly comparable with the others because they are only tested over nominal instances.

In Table 5.6 we introduce the results we obtained implementing the 1NN condition in our code. In particular, using our Eq. 5.3 with α set to 1 and $d = 0$, removing all context, we obtain an experimental setting equal to considering as the right answer the nearest vector to the target one. Our approach reveals directly compares with state of the art resources that use different techniques to build embeddings. In particular, our approach seems to improve over the results from all others approaches on SE02 and SE03 datasets, while ARES remains on top for SE07 and SE15 and SENSEMBERT$_{sup}$ wins in SE13 (although on a reduced problem, in that it only allows disambiguating nouns). Considering the concatenation of all the dataset SE-MACAROON seems the closest to human rating.

In Table 5.7 we report the results obtained testing on all the resources with the best configuration for SE-MACAROON, using Eq. 5.5 with $d = 20$: SE-MACAROON seems to be the only resource able to exploit both similarity between candidate sense and target word, and between candidate sense and context of the target word. In this specific and specially favorable setting SE-MACAROON consistently obtains the highest accuracy over all datasets.

The difference in performances of the other resources from those reported in Table 5.5 maybe due to differences in the implementation of the test. Our main strategy was that of employing a repository of senses along with their usage contexts, which is in spirit similar to a "memory", or to a "bag of embeddings" instead of a much complex but unique representation. Our experimental results seem to confirm that this approach may be beneficial to the WSD task, and allows attaining scores that directly compare with (if not improve over) other state of the art resources. Our strategy relies on the hypothesis that the target word may be compared with a variety of other candidates embeddings, amongst which we can find a similar one, instead of averaging all the information into one representation: in a sense, the present approach is intended to further de-conflate senses, and semantic blocks constituted by different contexts. This approach seems to be akin to the constructive rationale of attention-based embeddings. Furthermore, directly using the embeddings generated by the language model appears to enable the adoption of contextual word embeddings to weight similarity, which otherwise proved itself to be detrimental.

Finally, by examining the coverage reported in Table 5.8, we note that the coverage for both SENSEMBERT and SENSEMBERT$_{sup}$ is always close to the half of each benchmark, this is due to the fact that these resources have been built on nouns only. SE-MACAROON covers around 90% of each dataset, when not using Gemini addition, while SE-MACAROON with Gemini and both LMMS and LMMS-R together

TABLE 5.8: Coverage for each resource on the SemEval 17 English dataset. Reported figures express the absolute number of instances covered by the resource together with the percentage in square brackets [%].

Resource	SE02 n = 2282	SE03 n = 1850	SE07 n = 445	SE13 n = 1644	SE15 n = 1022	ALL n = 7253
LMMS	2282 [1.0]	1850 [1.0]	455 [1.0]	1644 [1.0]	1022 [1.0]	7253 [1.0]
SENSEMBERT	1262 [0.55]	1036 [0.56]	183 [0.4]	1644 [1.0]	633 [0.62]	4758 [0.66]
SENSEMBERT$_{sup}$	1262 [0.55]	1036 [0.56]	183 [0.4]	1644 [1.0]	633 [0.62]	4758 [0.66]
ARES	2282 [1.0]	1850 [1.0]	455 [1.0]	1644 [1.0]	1022 [1.0]	7253 [1.0]
LMMS-R	2282 [1.0]	1850 [1.0]	455 [1.0]	1644 [1.0]	1022 [1.0]	7253 [1.0]
SE-MACAROON	2052 [0.9]	1717 [0.93]	401 [0.88]	1432 [0.87]	874 [0.86]	6476 [0.89]
SE-MACAROON $_{Gemini}$	2282 [1.0]	1850 [1.0]	455 [1.0]	1644 [1.0]	1022 [1.0]	7253 [1.0]

with ARES, are able to deal with all instances in the evaluation framework. The low coverage of SENSEMBERT is reflected in lower recall scores: all the instances not covered are considered as same as errors when computing the recall. In fact, the recall for SENSEMBERT and SE-MACAROON without Gemini is lower when the coverage is under 90%, for example the recall for SE03 is 0.69 with a coverage of 93% while the recall for SE15 is 0.59 with a coverage of 0.86%.

Chapter 6

Conclusions

In this thesis we reported investigations in which different aspects of LMs played crucial roles. As mentioned, my PhD years were temporally aligned to one of the most prominent revolutions in the NLP field of the last decades: this fact gave me the opportunity to explore novel and vibrating technologies such as the Transformers architecture and the language models resulting from that computational device.

An overview on recent language models was provided in Chapter 2: language modeling task was introduced, also considering the underlying architectures, with specific emphasis on the Transformers architecture and on the architectural advancements leading to BERT and GPT-2, up to multimodal models such as OpenAI's GPT-4 and Google's Gemini.

In Chapter 3 Language Models were used in "classical" tasks, information extraction and error detection, relying on the embeddings generated from text. For the information extraction task, a BERT model was employed to deal with two different tasks: working on emergency room reports, the system was asked at first to identify the vehicles involved in road accidents, and then to detect the mean of transportation used by the patient at the time of the accident. These tasks were addressed as multi-label classification tasks, showing how the BERT representation is able to grasp elements in the text that are not simply identifiable through a string-matching approach. The validity of the results was also confirmed by some statistical insights on the road traffic accidents extracted. For the second classical task, a RoBERTa model was employed as a BIO sequence-to-sequence tagger, in order to identify Grammatical Errors in second-language (L2) acquisition: in particular, in this shared task systems were requested to identify tokens recognized as erroneous by human annotators. In this setting the system exploited transformers capability to detect anomalies in language in a multilingual setting, both when trained on the specific language, and on all languages at once.[1]

In Chapter 4 the generative features of autoregressive models were explored: the research question we addressed was whether the perplexity metrics can be interpreted as a semantic coherence marker, allowing to employ language models in the early detection of cognitive disorders. We showed how an intrinsic metrics (originally conceived to evaluate in how far a language model is able to predict real language) may be used to track the insurgence of a broad class of cognitive disorders affecting linguistic production. We considered both specific impairments — such as the Alzheimer Dementia— and a broad spectrum cognitive impairments with varied impairment degrees, i.e., mild, moderate, severe. In the experimentation we employed both English texts collected for research purposes, and more ecological conversations in Italian, widely varied experimental conditions were tested, yielding state of the art results. Our work proved to be helpful to assist clinicians and specialists in diagnosing the insurgence of dementia, which is a difficult and time-consuming task customarily involving a number of different assessments, such the neuropsychiatric evaluation (medical and neurologic history and examination), semistructured psychiatric interview, and neuropsychological assessments (Huff et al., 1987; Lopez et al., 1990; Becker et al., 1994). We also compared the scores obtained by employing different language models: the perplexity computed through simpler LMs may be a good option when language variability is reduced and/or training data are more consistent with test data. The remarkable differences between the work relying on the encoder and on the decoder modules of the Transformers architecture witness the wide capabilities of adaptation of such architecture (along with the related models) to different tasks. Additionally, the models we employed provided consistently valuable results both examining text produced in a controlled environment (by learners, through clinical reports, as description of an image) and truly ecological text, like the conversations between a patient and an operator.

SE-MACAROON vectors have been introduced in Chapter 5; these vectors were

[1]Our system resulted as the best performing system in the MultiGED-2023 shared task at NLP4CALL, `https://aclanthology.org/2023.nlp4call-1.1.pdf`.

obtained by collecting contextualized representations of words conveying a specific word sense. The hypothesis tested through this line of research was to verify whether such representations are precise enough to enable us to deal with the WSD task: as witnessed by the reported experimental results, the approach adopted closely compares with (at times overcoming) state of the art approaches. The proposed approach relies on two main pillars: exploiting the shared semantic space ensured by the sense grounding, and the adoption of an array of usage examples for the ⟨term, sense⟩ pair. We observed that these features allow obtaining a more comprehensive correspondence between the sentence at hand (containing the polysemous term) and the array of vector representations for the ⟨term, sense⟩ pair. Different from most contextualized sense embedding techniques, our proposal is to maintain, for each sense, a collection of separate word embeddings, avoiding the mix and the conflation of all possibly close meanings into a unique representation. Such design choice implements a kind of lexical memory, storing multiple representations of a word sense taken in its context. The underlying hypothesis is that keeping distinct representations for the same sense, possibly closer and more suited to several contexts of usage, may be beneficial for the disambiguation task. In this sense SE-MACAROON implements a preference for the meaning distinction, opposed to the meaning conflation approach; our design choices call for deepening our research towards cognitively plausible models of lexical access and retrieval in language production (Dell and O'Seaghdha, 1992), which will be addressed in future work. For the time present, SE-MACAROON vectors can be thought of as an attempt at building a semantic layer on top of language models combining distributional representations and more precise symbolic lexical knowledge.

Few final remarks follow on future work, and on how to carry on with undertaken research. Contextualized language models have gained a central role in many different NLP tasks and domains: one main axis of research points towards larger and larger models with less to no need for fine-tuning the previously acquired pre-trained models, as it earlier happened: in some cases larger models have been endowed with dedicated components for each different task or domain (Du et al., 2021). However, in some specific tasks large multilingual Transformer-based models are shown to be outperformed by their smaller variants, e.g. in predicting readers' eye fixations; such smaller variants thus indirectly proved to be more suited in analyzing cognitive phenomena such as lexical access and early semantic integration (De Varda and Marelli, 2023). Resources equipped with semantically grounded layers might be helpful in complementing such lighter model variants, and this will be the focus of our future work.

A second relevant line of research might be a novel attention to the evaluation frameworks of such models: for example, provided that large language models proved to be a valuable tool for many sorts of data augmentation in many fields ranging from computer vision and speech recognition (Perez and Wang, 2017) to grammatical error correction (Stahlberg and Kumar, 2021), there is still a relevant problem in the evaluation of their output. This will likely impact on benchmarks, as well, and on measuring the closeness between such benchmarks along with their associated downstream tasks and real-world applications and domains. In this setting, employing grounded word senses may be helpful in making interpretable the output of NLP systems and less opaque their generative behavior.

Finally, next steps will involve combining word senses: language models may be employed to make use of grammatical and syntactic information (Shen et al., 2017). For example, for extensive investigations on verbs along with their dependents (Kulmizev et al., 2020; Lakretz et al., 2021), to build applications to assist foreign language learners, and to assist specific sorts of cognitive impairments, such as in rehabilitation programs for individuals with deficits in language production, and more in general to gain insights on the overall knowledge structure underlying language models (Zhang et al., 2023).

The work done all throughout my PhD course, summarized in this dissertation, is linked to all these directions and to the increasingly rich possibilities that the Transformer architecture made and still makes newly available to researchers.

Appendix A

Sources of experimental material and detailed results

A.1 Material used in Experiment 1

The list of transcripts employed for training/fine tuning and testing, along with links to the `www.rev.com` platform can be found in the bundle containing the whole project, available at the URL `https://github.com/davidecolla/semantic_coherence_markers`.

A.2 Material used in Experiment 2

The list of transcripts employed for training/fine tuning and testing, along with links to the `www.rev.com` platform can be found in the bundle containing the whole project, available at the URL `https://github.com/davidecolla/semantic_coherence_markers`.

A.3 Statistics describing data and detailed results for Experiment 2

TABLE A.1: Figures describing the transcripts employed in Experiment 2: time duration, number of tokens, number of unique tokens (along with average number of tokens and average number of unique tokens) and type-token ratio (TTR) are reported for each such speech transcript.

Subject	Transcript	Duration	Tokens	Unique Tokens	AVG Tokens	AVG Unique Tokens	TTR
Joe Biden	I	0 : 32 : 23	4,647	1,074	6,315	1,343	0.21
	II	0 : 41 : 39	5,446	1,140			
	III	0 : 25 : 00	9,490	1,895			
	IV	0 : 43 : 36	6,801	1,381			
	V	0 : 34 : 05	5,211	1,226			
Donald Trump	I	1 : 17 : 37	15,200	1,967	15,051	1,185	0.13
	II	0 : 56 : 17	10,501	1,614			
	III	1 : 43 : 43	20,865	2,300			
	IV	1 : 13 : 01	14,056	1,945			
	V	1 : 18 : 19	14,806	1,896			
Barack Obama	I	0 : 56 : 39	5,594	1,479	5,957	1,271	0.21
	II	0 : 38 : 15	6,298	1,252			
	III	0 : 38 : 45	5,526	1,153			
	IV	0 : 45 : 55	6,981	1,312			
	V	0 : 36 : 07	5,390	1,159			
Bernie Sanders	I	0 : 35 : 33	4,164	969	4,458	1,046	0.23
	II	0 : 29 : 51	3,785	849			
	III	0 : 34 : 54	4,451	1,088			
	IV	0 : 43 : 27	5,387	1,039			
	V	0 : 44 : 46	4,501	1,286			
Bill Gates	I	0 : 35 : 53	3,503	944	2,514	812	0.32
	II	0 : 17 : 20	1,679	577			
	III	0 : 24 : 07	2,350	779			
	IV	0 : 22 : 04	2,152	744			
	V	0 : 30 : 07	2,896	1,018			
Nelson Mandela	I	0 : 40 : 17	3,844	1,113	6,403	1,410	0.22
	II	0 : 29 : 45	1,740	617			
	III	3 : 00 : 00	15,682	2,702			
	IV	1 : 43 : 21	7,741	1,654			
	V	0 : 40 : 16	3,020	963			
Martin Luther King	I	0 : 42 : 51	5,197	1,102	6,508	1,379	0.21
	II	0 : 46 : 56	6,471	1,315			
	III	0 : 43 : 48	6,287	1,456			
	IV	0 : 40 : 38	8,256	1,697			
	V	0 : 47 : 54	6,332	1,324			
Boris Johnson	I	0 : 51 : 42	4,397	1,123	3,202	943	0.29
	II	0 : 20 : 35	2,758	764			
	III	0 : 17 : 47	1,960	659			
	IV	0 : 17 : 00	2,375	896			
	V	0 : 38 : 22	4,530	1,273			

A.4 Detailed results for Experiment 3

TABLE A.2: Detailed results obtained in Experiment 2 (between subjects reliability): each sub-table reports results for N-gram models. For each experiment we report perplexity scores along with their standard deviations. More specifically, we report the scores obtained by employing 2-grams to 5-grams models. Each row reports the scores obtained through the LM trained on speeches by the subject in the first column and tested on the other speakers.

2-grams

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	262.42	4.03	223.76	6.94	199.50	5.86	209.92	17.74	181.64	16.15	191.22	13.16	197.47	26.94
D. Trump	276.71	18.05	—	—	268.72	7.02	230.47	4.43	241.55	23.36	205.11	21.21	219.05	19.62	225.97	36.59
B. Obama	242.32	14.15	281.05	3.99	—	—	202.07	5.96	219.87	17.27	188.89	18.43	198.19	16.07	205.93	29.94
B. Sanders	202.26	9.93	231.57	4.33	197.19	5.50	—	—	191.61	10.02	166.92	14.74	172.84	11.60	179.69	18.10
B. Gates	221.97	11.75	253.93	3.69	215.63	5.47	197.46	3.48	—	—	177.97	16.40	186.61	12.56	194.60	24.05
N. Mandela	215.75	14.05	245.86	5.17	200.55	8.16	192.64	7.10	199.97	7.89	—	—	183.76	16.17	190.32	24.37
M. L. King	232.65	12.95	267.86	3.33	225.78	5.87	199.83	6.48	215.69	14.60	187.93	18.35	—	—	201.88	26.07
B. Johnson	222.69	12.78	253.65	3.46	215.32	7.38	197.24	6.29	207.91	13.94	179.03	17.05	187.25	12.98	—	—

3-grams

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	406.05	8.04	335.23	22.01	262.33	8.07	263.84	32.53	210.22	17.30	247.03	37.22	244.37	37.09
D. Trump	454.56	40.38	—	—	461.33	37.33	332.98	8.51	329.39	50.45	250.27	25.36	310.56	63.55	301.00	54.22
B. Obama	358.21	27.74	445.25	9.07	—	—	266.61	14.83	278.93	33.74	217.98	19.41	257.84	44.90	256.96	40.79
B. Sanders	271.69	16.91	323.58	5.99	273.11	14.96	—	—	228.20	19.08	185.62	14.71	209.38	26.40	212.36	24.03
B. Gates	310.96	21.20	377.88	6.84	316.11	19.48	252.86	6.72	—	—	202.51	16.94	235.63	32.85	236.38	32.09
N. Mandela	279.08	21.42	330.54	7.52	265.66	10.72	233.46	9.95	235.10	15.99	—	—	219.70	29.87	220.96	30.05
M. L. King	333.91	24.66	410.89	6.77	339.45	23.40	257.78	13.92	269.14	28.06	215.84	19.19	—	—	248.40	35.49
B. Johnson	301.58	20.95	357.49	5.50	301.67	15.95	247.23	6.66	250.34	25.24	201.36	17.66	230.76	28.22	—	—

4-grams

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	511.58	10.26	408.86	29.02	313.72	8.29	308.91	44.85	239.39	17.26	294.40	49.96	283.25	47.23
D. Trump	595.73	59.19	—	—	601.17	52.71	425.11	11.01	407.96	74.91	297.95	28.24	393.33	91.56	366.84	72.73
B. Obama	434.45	36.32	553.74	10.74	—	—	314.58	19.09	325.33	46.01	247.72	19.14	305.16	58.41	297.27	51.33
B. Sanders	316.38	20.87	385.39	6.63	318.39	18.59	—	—	257.53	25.42	205.20	13.84	239.21	32.71	238.26	30.06
B. Gates	371.64	27.30	463.54	8.60	378.35	24.81	296.05	7.08	—	—	227.73	16.36	275.74	41.42	270.49	40.51
N. Mandela	319.85	24.56	387.17	8.18	308.12	13.59	264.24	10.35	263.71	21.80	—	—	249.01	34.04	245.99	35.71
M. L. King	403.33	32.13	509.82	8.58	409.59	29.81	302.47	17.95	312.81	39.24	244.41	18.76	—	—	286.18	45.07
B. Johnson	357.56	26.52	435.98	7.46	359.13	20.68	287.57	7.20	286.85	33.98	224.76	17.14	268.16	35.35	—	—

5-grams

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	624.81	13.72	491.81	35.69	369.76	9.27	360.01	58.55	273.18	18.93	346.38	59.97	327.28	58.93
D. Trump	750.16	76.77	—	—	765.22	68.36	529.26	14.38	499.48	103.94	354.69	34.56	486.85	115.52	444.11	95.07
B. Obama	516.13	44.54	671.90	14.05	—	—	367.86	24.24	378.35	59.87	282.25	20.26	357.79	69.76	343.16	63.57
B. Sanders	364.42	25.11	452.64	8.10	369.41	22.24	—	—	290.97	32.57	227.70	13.57	272.37	38.10	267.51	37.06
B. Gates	438.15	33.74	558.68	11.50	449.35	30.61	344.39	7.59	—	—	256.98	16.87	320.80	49.22	309.31	50.31
N. Mandela	367.01	28.55	452.77	9.31	357.29	17.00	299.51	10.97	296.75	28.68	—	—	282.82	39.02	274.89	42.54
M. L. King	478.32	39.58	618.46	11.63	489.60	36.37	352.03	23.17	362.62	51.94	277.48	19.59	—	—	329.21	56.26
B. Johnson	418.83	32.46	523.31	10.17	424.61	25.77	332.46	7.98	328.72	43.92	252.02	17.48	310.22	41.90	—	—

TABLE A.3: Detailed results obtained in Experiment 2 (between subjects reliability): each sub-table reports results for a GPT-2-based language model (differences stem from the number of fine tuning epochs employed to acquire each such model). For each experiment we report perplexity scores along with their standard deviations. Each row reports the scores obtained through the LM trained on speeches by the subject in the first column and tested on the other speakers.

GPT-2 5 **epochs**

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	22.48	0.90	21.28	1.94	26.32	3.74	18.39	2.58	29.23	2.41	28.17	3.05	42.49	7.85
D. Trump	23.27	1.72	—	—	20.42	2.37	26.39	3.84	17.96	2.24	31.01	2.43	28.48	3.15	42.93	8.04
B. Obama	24.61	1.79	22.58	0.98	—	—	26.37	3.75	18.28	2.56	29.39	2.36	28.10	3.07	43.03	7.87
B. Sanders	27.47	1.84	25.60	1.16	23.40	1.83	—	—	19.92	2.83	29.61	2.51	29.54	3.32	46.23	8.42
B. Gates	25.16	1.73	23.26	0.95	21.82	1.96	26.84	3.83	—	—	28.71	2.32	27.96	2.92	43.30	7.93
N. Mandela	28.54	1.97	26.38	1.25	24.45	1.89	28.62	3.84	19.63	2.51	—	—	28.63	3.12	45.80	8.45
M. L. King	25.39	1.76	23.51	1.09	21.68	1.76	26.21	3.63	18.45	2.46	28.03	2.50	—	—	42.86	7.86
B. Johnson	26.20	1.79	24.75	0.99	22.76	1.90	27.30	3.80	19.34	2.84	28.93	2.40	28.62	3.01	—	—

GPT-2 10 **epochs**

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	21.24	0.87	20.15	1.94	25.72	3.84	17.84	2.51	30.23	2.35	28.11	3.18	41.71	7.89
D. Trump	23.46	1.88	—	—	20.49	2.79	28.04	4.17	18.33	2.26	34.65	2.64	30.39	3.74	45.40	8.72
B. Obama	23.60	1.89	21.41	0.97	—	—	26.10	3.92	17.84	2.49	30.54	2.32	28.41	3.29	43.01	8.01
B. Sanders	26.06	1.65	24.30	1.11	22.39	1.68	—	—	19.27	2.66	29.25	2.38	28.66	3.30	44.52	8.13
B. Gates	24.34	1.67	22.31	0.96	20.99	1.94	26.41	3.94	—	—	29.53	2.21	28.04	3.07	43.13	8.07
N. Mandela	28.78	1.99	26.37	1.31	24.54	1.87	28.60	3.98	19.65	2.40	—	—	28.74	3.09	46.15	8.66
M. L. King	25.22	1.78	22.88	1.19	21.33	1.70	26.13	3.66	18.27	2.32	28.54	2.54	—	—	43.32	8.20
B. Johnson	25.03	1.61	24.00	0.88	21.89	1.77	26.54	3.77	18.85	2.87	28.91	2.33	27.93	2.90	—	—

GPT-2 20 **epochs**

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	20.60	0.94	19.69	2.11	26.36	4.10	17.79	2.42	32.29	2.42	29.16	3.58	42.70	8.33
D. Trump	25.38	2.42	—	—	21.91	3.62	32.55	4.86	20.23	2.91	43.10	2.76	35.18	5.08	52.92	10.60
B. Obama	24.10	2.24	21.44	1.07	—	—	27.46	4.25	18.25	2.51	33.17	2.50	30.18	3.81	45.92	8.77
B. Sanders	26.19	1.59	24.29	1.18	22.30	1.46	—	—	19.36	2.58	30.44	2.40	29.34	3.59	45.62	8.44
B. Gates	24.90	1.92	22.21	1.08	21.18	2.16	27.42	4.19	—	—	31.54	2.13	29.53	3.57	45.88	8.75
N. Mandela	29.64	2.11	26.70	1.40	25.15	1.99	29.26	4.18	20.06	2.31	—	—	29.64	3.21	48.01	9.48
M. L. King	27.10	2.09	23.72	1.48	22.55	1.90	28.16	3.88	19.34	2.28	31.31	2.83	—	—	48.12	9.61
B. Johnson	24.68	1.59	24.20	0.82	21.69	1.75	26.68	4.02	18.92	3.16	29.87	2.34	28.31	3.02	—	—

GPT-2 30 **epochs**

Subject	J. Biden		D. Trump		B. Obama		B. Sanders		B. Gates		N. Mandela		M. L. King		B. Johnson	
	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev	PPL	stdev
J. Biden	—	—	21.18	1.03	20.14	2.36	27.98	4.49	18.42	2.53	35.04	2.59	31.11	4.09	45.47	9.18
D. Trump	28.07	3.07	—	—	24.07	4.63	37.73	5.76	22.67	3.76	52.84	3.02	41.10	6.74	62.44	13.05
B. Obama	26.10	2.71	23.14	1.24	—	—	30.12	4.73	19.69	2.78	37.07	2.85	33.41	4.54	51.18	9.91
B. Sanders	27.16	1.73	24.88	1.32	22.91	1.44	—	—	19.93	2.55	31.91	2.48	30.70	3.95	48.29	9.21
B. Gates	26.83	2.35	23.47	1.31	22.57	2.55	29.68	4.58	—	—	34.83	2.11	32.38	4.28	51.16	9.75
N. Mandela	31.40	2.33	27.94	1.53	26.47	2.15	30.84	4.47	21.01	2.43	—	—	31.60	3.59	51.74	10.70
M. L. King	30.26	2.50	25.79	1.82	24.78	2.25	31.23	4.33	21.32	2.51	36.00	3.20	—	—	55.62	11.67
B. Johnson	25.43	1.71	25.17	0.82	22.33	1.79	27.75	4.26	19.60	3.50	31.33	2.51	29.53	3.20	—	—

TABLE A.4: Detailed results for Experiment 3. The table reports Accuracy (Acc.) scores, Precision (P), Recall (R) and F1 for both tasks of identifying AD and Control subjects. The rightmost column reports the harmonic mean (HM) of the accuracy, F1 score on the AD and C classes. Best results are marked in boldface.

Model		Acc.	Dementia (AD)			Control (C)			HM(acc,F1$_{AD}$,F1$_C$)
			P	R	F1	P	R	F1	
2-grams	$\overline{P}_C$	0.44	0.41	0.21	0.28	0.46	0.69	0.55	0.39
	$\overline{P}_{AD}$	0.55	0.54	0.75	0.63	0.57	0.34	0.42	0.52
	$\overline{D}$	0.67	0.68	0.66	0.67	0.66	0.68	0.67	0.67
	$\overline{D}^*$	**0.93**	0.99	0.87	**0.92**	0.88	0.99	**0.93**	**0.93**
3-grams	$\overline{P}_C$	0.43	0.40	0.22	0.28	0.44	0.65	0.53	0.39
	$\overline{P}_{AD}$	0.56	0.55	0.70	0.62	0.57	0.41	0.47	0.54
	$\overline{D}$	0.74	0.76	0.71	0.74	0.72	0.77	0.75	0.74
	$\overline{D}^*$	**0.91**	1.00	0.83	**0.91**	0.85	1.00	**0.92**	**0.91**
4-grams	$\overline{P}_C$	0.42	0.38	0.23	0.29	0.43	0.61	0.51	0.38
	$\overline{P}_{AD}$	0.54	0.54	0.65	0.59	0.54	0.43	0.48	0.53
	$\overline{D}$	0.76	0.81	0.70	0.75	0.73	0.82	0.77	0.76
	$\overline{D}^*$	**0.89**	1.00	0.78	**0.88**	0.81	1.00	**0.90**	**0.89**
5-grams	$\overline{P}_C$	0.42	0.38	0.23	0.29	0.43	0.61	0.51	0.38
	$\overline{P}_{AD}$	0.52	0.53	0.62	0.57	0.52	0.42	0.46	0.52
	$\overline{D}$	0.77	0.86	0.66	0.75	0.72	0.89	0.80	0.77
	$\overline{D}^*$	**0.89**	1.00	0.79	**0.88**	0.82	1.00	**0.90**	**0.89**
GPT-2 5 epochs	$\overline{P}_C$	0.65	0.64	0.70	0.67	0.66	0.59	0.62	0.65
	$\overline{P}_{AD}$	0.38	0.42	0.58	0.49	0.29	0.18	0.22	0.33
	$\overline{D}$	**0.71**	0.76	0.62	**0.69**	0.67	0.80	**0.73**	**0.71**
	$\overline{D}^*$	0.49	0.50	0.09	0.15	0.49	0.91	0.64	0.30
GPT-2 10 epochs	$\overline{P}_C$	0.78	0.70	0.99	0.82	0.98	0.57	0.72	0.77
	$\overline{P}_{AD}$	0.62	0.63	0.58	0.61	0.60	0.65	0.62	0.62
	$\overline{D}$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**
	$\overline{D}^*$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**
GPT-2 20 epochs	$\overline{P}_C$	0.81	0.73	1.00	0.84	1.00	0.61	0.76	0.80
	$\overline{P}_{AD}$	0.78	0.91	0.64	0.75	0.71	0.93	0.81	0.78
	$\overline{D}$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**
	$\overline{D}^*$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**
GPT-2 30 epochs	$\overline{P}_C$	0.81	0.73	1.00	0.84	1.00	0.61	0.76	0.80
	$\overline{P}_{AD}$	0.81	0.96	0.65	0.78	0.73	0.97	0.83	0.80
	$\overline{D}$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**
	$\overline{D}^*$	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	**1.00**

Bibliography

Agirre, Eneko, Oier López de Lacalle, and Aitor Soroa (Mar. 2014). "Random Walks for Knowledge-Based Word Sense Disambiguation". In: *Computational Linguistics* 40.1, pp. 57–84. DOI: 10.1162/COLI_a_00164. URL: https://aclanthology.org/J14-1003.

Akbik, Alan, Duncan Blythe, and Roland Vollgraf (2018). "Contextual string embeddings for sequence labeling". In: *Proceedings of the 27th international conference on computational linguistics*, pp. 1638–1649.

Aldinucci, Marco, Stefano Bagnasco, Stefano Lusso, Paolo Pasteris, Sergio Rabellino, and Sara Vallero (2017). "OCCAM: a flexible, multi-purpose and extendable HPC cluster". In: *Journal of Physics: Conference Series* 898.8, pp. 082039–082047.

AlMamlook, Rabia Emhamed, Keneth Morgan Kwayu, Maha Reda Alkasisbeh, and Abdulbaset Ali Frefer (2019). "Comparison of machine learning algorithms for predicting traffic accident severity". In: *2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT)*. IEEE, pp. 272–276.

Anil, Rohan, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. (2023). "Gemini: a family of highly capable multimodal models". In: *arXiv preprint arXiv:2312.11805*.

Assi, Khaled (2020). "Traffic crash severity prediction—A synergy by hybrid principal component analysis and machine learning models". In: *International journal of environmental research and public health* 17.20, p. 7598.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Bao, Wentao, Qi Yu, and Yu Kong (2020). "Uncertainty-based traffic accident anticipation with spatio-temporal relational learning". In: *Proceedings of the 28th ACM International Conference on Multimedia*, pp. 2682–2690.

Becker, James T, François Boiler, Oscar L Lopez, Judith Saxton, and Karen L McGonigle (1994). “The natural history of Alzheimer’s disease: description of study cohort and accuracy of diagnosis”. In: *Archives of Neurology* 51.6, pp. 585–594.

Bedi, Gillinder, Facundo Carrillo, Guillermo A Cecchi, Diego Fernández Slezak, Mariano Sigman, Natália B Mota, Sidarta Ribeiro, Daniel C Javitt, Mauro Copelli, and Cheryl M Corcoran (2015). “Automated analysis of free speech predicts psychosis onset in high-risk youths”. In: *npj Schizophrenia* 1.1, pp. 1–7.

Bell, Samuel, Helen Yannakoudakis, and Marek Rei (Aug. 2019). “Context is Key: Grammatical Error Detection with Contextual Word Representations”. In: *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Florence, Italy: Association for Computational Linguistics, pp. 103–115. DOI: 10.18653/v1/W19-4410. URL: https://aclanthology.org/W19-4410.

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). “Learning long-term dependencies with gradient descent is difficult”. In: *IEEE transactions on neural networks* 5.2, pp. 157–166.

Benvenuti, Nicola, Andrea Bolioli, Alessio Bosca, Alessandro Mazzei, Pietro Vigorelli, et al. (2020). “The" Corpus Anchise 320" and the analysis of conversations between healthcare workers and people with dementia”. In: *Proc. of Seventh Italian Conference on Computational Linguistics*. CEUR, pp. 1–6.

Boer, Janna N de, Sanne G Brederoo, Alban E Voppel, and Iris EC Sommer (2020). “Anomalies in language as a biomarker for schizophrenia”. In: *Current opinion in psychiatry* 33.3, pp. 212–218.

Bojanowski, Piotr, Édouard Grave, Armand Joulin, and Tomáš Mikolov (2017). “Enriching Word Vectors with Subword Information”. In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146.

Bokaba, Tebogo, Wesley Doorsamy, and Babu Sena Paul (2022). “Comparative study of machine learning classifiers for modelling road traffic accidents”. In: *Applied Sciences* 12.2, p. 828.

Bond, Francis and Kyonghee Paik (2012). “A survey of wordnets and their licenses”. In: *Small* 8.4, p. 5.

Boyd, Adriane (2018). “Using Wikipedia edits in low resource grammatical error correction”. In: *Proceedings of the 2018 EMNLP Workshop W-NUT: The 4th Workshop on Noisy User-generated Text*, pp. 79–84.

Brown, Tom B. et al. (2020). “Language models are few-shot learners”. In: *Advances in neural information processing systems* 33, pp. 1877–1901.

Bryant, Christopher, Mariano Felice, Øistein E. Andersen, and Ted Briscoe (Aug. 2019). “The BEA-2019 Shared Task on Grammatical Error Correction”. In: *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Florence, Italy: Association for Computational Linguistics, pp. 52–75. DOI: 10.18653/v1/W19-4406. URL: https://aclanthology.org/W19-4406.

Bryant, Christopher, Zheng Yuan, Muhammad Reza Qorib, Hannan Cao, Hwee Tou Ng, and Ted Briscoe (2022). “Grammatical Error Correction: A Survey of the State of the Art”. In: *arXiv preprint arXiv:2211.05166*.

Cabana, Álvaro, Juan C Valle-Lisboa, Brita Elvevåg, and Eduardo Mizraji (2011). “Detecting order–disorder transitions in discourse: Implications for schizophrenia”. In: *Schizophrenia Research* 131.1-3, pp. 157–164.

Camacho-Collados, Jose and Mohammad Taher Pilehvar (2018). “From word to sense embeddings: A survey on vector representations of meaning”. In: *Journal of Artificial Intelligence Research* 63, pp. 743–788.

Camacho-Collados, José, Mohammad Taher Pilehvar, and Roberto Navigli (2015). “NASARI: a novel approach to a semantically-aware representation of items”. In: *Proceedings of NAACL*, pp. 567–577.

Chang, Yupeng, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. (2023). “A survey on evaluation of large language models”. In: *ACM Transactions on Intelligent Systems and Technology*.

Chong, Miao, Ajith Abraham, and Marcin Paprzycki (2005). “Traffic accident analysis using machine learning paradigms”. In: *Informatica* 29.1.

Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. (2023). “Palm: Scaling language modeling with pathways”. In: *Journal of Machine Learning Research* 24.240, pp. 1–113.

Christiano, Paul F, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei (2017). “Deep reinforcement learning from human preferences”. In: *Advances in neural information processing systems* 30.

Church, Kenneth Ward, Zeyu Chen, and Yanjun Ma (2021). “Emerging trends: A gentle introduction to fine-tuning”. In: *Natural Language Engineering* 27.6, pp. 763–778.

City of Montreal (Accessed: 10-10-2022). *Montreal Veichle Collisions*. http://donnees.ville.montreal.qc.ca/dataset/collisions-routieres.

Clark, Kevin, Minh-Thang Luong, Quoc V Le, and Christopher D Manning (2020). “ELECTRA: Pre-training text encoders as discriminators rather than generators”. In: *arXiv preprint arXiv:2003.10555*.

Clavié, Benjamin, Alexandru Ciceu, Frederick Naylor, Guillaume Soulié, and Thomas Brightwell (2023). “Large Language Models in the Workplace: A Case Study on Prompt Engineering for Job Type Classification”. In: *International Conference on Applications of Natural Language to Information Systems*. Springer, pp. 3–17.

Cohen, Trevor and Serguei Pakhomov (July 2020). “A Tale of Two Perplexities: Sensitivity of Neural Language Models to Lexical Retrieval Deficits in Dementia of the Alzheimer’s Type”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 1946–1957. DOI: 10.18653/v1/2020.acl-main.176. URL: https://aclanthology.org/2020.acl-main.176.

Cohn, Trevor (2003). “Performance metrics for word sense disambiguation”. In: *Proceedings of the Australasian Language Technology Workshop 2003*, pp. 86–93.

Çokal, Derya, Gabriel Sevilla, William Stephen Jones, Vitor Zimmerer, Felicity Deamer, Maggie Douglas, Helen Spencer, Douglas Turkington, Nicol Ferrier, Rosemary Varley, et al. (2018). “The language profile of formal thought disorder”. In: *npj Schizophrenia* 4.1, pp. 1–8.

Colla, Davide, Matteo Delsanto, Marco Agosto, Benedetto Vitiello, and Daniele P Radicioni (2022). “Semantic coherence markers: The contribution of perplexity metrics”. In: *Artificial Intelligence in Medicine* 134, p. 102393.

Colla, Davide, Matteo Delsanto, and Elisa Di Nuovo (2023). “ELICODE at MultiGED2023: fine-tuning XLM-RoBERTa for multilingual grammatical error detection”. In: *LINKÖPING ELECTRONIC CONFERENCE PROCEEDINGS*. Vol. 197. Linköping University Electronic Press, pp. 24–34.

Colla, Davide, Enrico Mensa, and Daniele P Radicioni (2020). “LessLex: Linking multilingual embeddings to SenSe representations of LEXical items”. In: *Computational Linguistics* 46.2, pp. 289–333.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (2019). “Unsupervised Cross-lingual Representation Learning at Scale”. In.

Covington, Michael A, Congzhou He, Cati Brown, Lorina Naçi, Jonathan T McClain, Bess Sirmon Fjordbak, James Semple, and John Brown (2005). “Schizophrenia and the structure of language: the linguist’s view”. In: *Schizophrenia research* 77.1, pp. 85–98.

Das, Subasish, Songjukta Datta, Hamsa Abbas Zubaidi, and Ihsan Ali Obaid (2021). “Applying interpretable machine learning to classify tree and utility pole related crash injury types”. In: *IATSS research* 45.3, pp. 310–316.

De Mattei, Lorenzo, Michele Cafagna, Felice Dell’Orletta, Malvina Nissim, and Marco Guerini (2020). “Geppetto carves italian into a language model”. In: *arXiv preprint arXiv:2004.14253*.

De Varda, Andrea and Marco Marelli (2023). “Scaling in cognitive modelling: A multilingual approach to human reading times”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 139–149.

Dell, Gary S and Padraig G O’Seaghdha (1992). “Stages of lexical access in language production”. In: *Cognition* 42.1-3, pp. 287–314.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805*.

Di Nuovo, Elisa (2022). *VALICO-UD: annotating an Italian learner corpus*. Doctoral Thesis. University of Genoa and University of Turin.

Docherty, Nancy M, Maddalena DeRosa, and Nancy C Andreasen (1996). “Communication disturbances in schizophrenia and mania”. In: *Archives of General Psychiatry* 53.4, pp. 358–364.

Du, Nan, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. (2021). “GLaM: Efficient Scaling of Language Models with Mixture-of-Experts”. In: *arXiv preprint arXiv:2112.06905*.

Edmonds, Philip and Scott Cotton (2001). “Senseval-2: overview”. In: *Proceedings of SENSEVAL-2 Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pp. 1–5.

Elman, Jeffrey L (1990). “Finding structure in time”. In: *Cognitive science* 14.2, pp. 179–211.

Elvevåg, Brita, Peter W Foltz, Daniel R Weinberger, and Terry E Goldberg (2007). “Quantifying incoherence in speech: an automated methodology and novel application to schizophrenia”. In: *Schizophrenia research* 93.1-3, pp. 304–316.

Folstein, Marshal F, Susan E Folstein, and Paul R McHugh (1975). ““Mini-mental state”: a practical method for grading the cognitive state of patients for the clinician”. In: *Journal of psychiatric research* 12.3, pp. 189–198.

Frankenberg, Claudia, Jochen Weiner, Tanja Schultz, Maren Knebel, Christina Degen, Hans-W Wahl, and Johannes Schroeder (2019). “Perplexity –a new predictor of cognitive changes in spoken language?– results of the Interdisciplinary Longitudinal Study on Adult Development and Aging (ILSE)”. In: *Linguistics Vanguard* 5.2, pp. 1–10.

Fritsch, Julian, Sebastian Wankerl, and Elmar Nöth (2019). “Automatic diagnosis of Alzheimer’s disease using neural network language models”. In: *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5841–5845.

Gale, William A and Kenneth W Church (1994). “What’s wrong with adding one”. In: *Corpus-based research into language: In honour of Jan Aarts*, pp. 189–200.

Gehring, Jonas, Michael Auli, David Grangier, and Yann N Dauphin (2016). “A convolutional encoder model for neural machine translation”. In: *arXiv preprint arXiv:1611.02344*.

Goikoetxea, Josu, Aitor Soroa, and Eneko Agirre (June 2018). "Bilingual Embeddings with Random Walks over Multilingual Wordnets". In: *Knowledge-Based Systems* 150.C, pp. 218–230. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2018.03.017.

Goldberg, Yoav (2017). "Neural network methods for natural language processing". In: *Synthesis Lectures on Human Language Technologies* 10.1, pp. 1–309.

Goodglass, Harold and Edith Kaplan (1983). *Boston diagnostic aphasia examination booklet*. Lea & Febiger.

Government of Canada (Accessed: 10-10-2022). *National Road Network*. https://open.canada.ca/data/en/dataset/3d282116-e556-400c-9306-ca1a3cada77f.

Granger, Sylviane (1998). "The computer learner corpus: a versatile new source of data for SLA research". In: *Learner English on computer*. Routledge, pp. 3–18.

Granger, Sylviane, Maïté Dupont, Fanny Meunier, and Magali Paquot (2020). "International Corpus of Learner English. Version 3 (Handbook and web interface)". In: *Louvain-la-Neuve: Presses Universitaires de Louvain*, p. 134.

Grootendorst, Maarten (2020). *KeyBERT: Minimal keyword extraction with BERT.* Version v0.3.0. DOI: 10.5281/zenodo.4461265. URL: https://doi.org/10.5281/zenodo.4461265.

Gutierrez-Osorio, Camilo and César Pedraza (2020). "Modern data sources and techniques for analysis and forecast of road accidents: A review". In: *Journal of traffic and transportation engineering (English edition)* 7.4, pp. 432–446.

Haber, Janosch and Massimo Poesio (2023). "Polysemy—Evidence from Linguistics, Behavioral Science, and Contextualized Language Models". In: *Computational Linguistics* 50.1, pp. 351–417.

Harman, Donna (1993). "Overview of the first TREC conference". In: *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 36–47.

Harris, Zellig S (1954). "Distributional structure". In: *Word* 10.2-3, pp. 146–162.

Helms, Joel R (2009). *Mathematics for Health Sciences: A Comprehensive Approach*. Cengage Learning.

Hindle, Donald (1990). "Noun classification from predicate-argument structures". In: *28th Annual meeting of the Association for Computational Linguistics*, pp. 268–275.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory". In: *Neural computation* 9.8, pp. 1735–1780.

Huang, Eric H, Richard Socher, Christopher D Manning, and Andrew Y Ng (2012). "Improving word representations via global context and multiple word prototypes". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, pp. 873–882.

Huff, F Jacob, JT Becker, SH Belle, RD Nebes, AL Holland, and F Boller (1987). "Cognitive deficits and clinical diagnosis of Alzheimer's disease". In: *Neurology* 37.7, pp. 1119–1119.

Iacobacci, Ignacio and Roberto Navigli (2019). "LSTMEmbed: Learning Word and Sense Representations from a Large Semantically Annotated Corpus with Long Short-Term Memories". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1685–1695.

Iacobacci, Ignacio, Mohammad Taher Pilehvar, and Roberto Navigli (2015). "Sensembed: Learning sense embeddings for word and relational similarity". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Vol. 1, pp. 95–105.

James, Carl (1998). *Errors in language learning and use*. Pearson Educational Limited.

Jawahar, Ganesh, Benoît Sagot, and Djamé Seddah (2019). "What does BERT learn about the structure of language?" In: *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Jurafsky, Dan and James H Martin (2014). *Speech and language processing. Vol. 3*. Prentice Hall.

Kaneko, Masahiro and Mamoru Komachi (2019). "Multi-head multi-layer attention to deep language representations for grammatical error detection". In: *Computación y Sistemas* 23.3, pp. 883–891.

Kilgarriff, Adam (2001). "Comparing corpora". In: *International journal of corpus linguistics* 6.1, pp. 97–133.

Kilgarriff, Adam, Vít Baisa, Jan Bušta, Miloš Jakubíček, Vojtěch Kovář, Jan Michelfeit, Pavel Rychlỳ, and Vít Suchomel (2014). "The Sketch Engine: ten years on". In: *Lexicography* 1.1, pp. 7–36.

Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel (2014). "Multimodal neural language models". In: *International conference on machine learning*. PMLR, pp. 595–603.

Kneser, Reinhard and Hermann Ney (1995). "Improved backing-off for m-gram language modeling". In: *1995 international conference on acoustics, speech, and signal processing*. Vol. 1. IEEE, pp. 181–184.

Kudo, Taku and John Richardson (2018). "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing". In: *arXiv preprint arXiv:1808.06226*.

Kulmizev, Artur, Vinit Ravishankar, Mostafa Abdou, and Joakim Nivre (2020). "Do neural language models show preferences for syntactic formalisms?" In: *arXiv preprint arXiv:2004.14096*.

Kuzmenko, Elizaveta and Andrey Kutuzov (Nov. 2014). "Russian Error-Annotated Learner English Corpus: a Tool for Computer-Assisted Language Learning". In: *Proceedings of the third workshop on NLP for computer-assisted language learning*. Uppsala, Sweden: LiU Electronic Press, pp. 87–97. URL: https://aclanthology.org/W14-3507.

Lakretz, Yair, Dieuwke Hupkes, Alessandra Vergallito, Marco Marelli, Marco Baroni, and Stanislas Dehaene (2021). "Mechanisms for handling nested dependencies in neural-network language models and humans". In: *Cognition* 213, p. 104699.

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). "Albert: A lite BERT for self-supervised learning of language representations". In: *arXiv preprint arXiv:1909.11942*.

Landauer, Thomas K, Peter W Foltz, and Darrell Laham (1998). "An introduction to latent semantic analysis". In: *Discourse Processes* 25.2-3, pp. 259–284.

Lanzoni, Alessandro, Andrea Fabbo, Donatella Basso, Patrizia Pedrazzini, Elena Bortolomiol, Marc Jones, and Omar Cauli (2018). "Interventions aimed to increase independence and well-being in patients with Alzheimer's disease: Review of some interventions in the Italian context". In: *Neurology, Psychiatry and Brain Research* 30, pp. 137–143.

Larson, Molly K, Elaine F Walker, and Michael T Compton (2010). "Early signs, diagnosis and therapeutics of the prodromal phase of schizophrenia and related psychotic disorders". In: *Expert review of neurotherapeutics* 10.8, pp. 1347–1359.

Lennon, Paul (1991). "Error: Some problems of definition, identification, and distinction". In: *Applied linguistics* 12.2, pp. 180–196.

Li, Zhibin, Pan Liu, Wei Wang, and Chengcheng Xu (2012). "Using support vector machine models for crash injury severity analysis". In: *Accident Analysis & Prevention* 45, pp. 478–486.

Liljequist, David, Britt Elfving, and Kirsti Skavberg Roaldsen (2019). "Intraclass correlation–A discussion and demonstration of basic features". In: *PloS one* 14.7, e0219854.

Little, David (2006). "The Common European Framework of Reference for Languages: Content, purpose, origin, reception and impact". In: *Language Teaching* 39.3, pp. 167–190.

Liu, Peter J, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer (2018). "Generating Wikipedia by summarizing long sequences". In: *arXiv preprint arXiv:1801.10198*.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692*.

Lopez, Oscar L, AA Swihart, James T Becker, OM Reinmuth, CF Reynolds, DL Rezek, and FL Daly (1990). "Reliability of NINCDS-ADRDA clinical criteria for the diagnosis of Alzheimer's disease". In: *Neurology* 40.10, pp. 1517–1517.

Loureiro, Daniel and Alípio Jorge (July 2019). "Language Modelling Makes Sense: Propagating Representations through WordNet for Full-Coverage Word Sense Disambiguation". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 5682–5691. DOI: 10.18653/v1/P19-1569. URL: https://aclanthology.org/P19-1569.

Loureiro, Daniel, Alípio Mário Jorge, and Jose Camacho-Collados (2022). "LMMS reloaded: Transformer-based sense embeddings for disambiguation and beyond". In: *Artificial Intelligence*, p. 103661.

Lüdeling, Anke and Hagen Hirschmann (2015). "Error annotation systems". In: *The Cambridge handbook of learner corpus research*. Ed. by Sylviane Granger, Gaëtanelle Gilquin, and Fanny Meunier. Cambridge: Cambridge University Press, pp. 135–157.

Lyons, Ronan, Rupert Kisse, and Wim Rogmans (2015). *EU-Injury database Introduction to the functioning of the Injury Database (IDB)*. https://bit.ly/37FAKaB.

MacWhinney, Brian (2014). *The CHILDES project: Tools for analyzing talk, Volume II: The database*. Psychology Press.

— (2017). "Tools for analyzing talk part 1: The CHAT transcription format". In: *Carnegie*, pp. 1–115.

Mancini, Massimiliano, Jose Camacho-Collados, Ignacio Iacobacci, and Roberto Navigli (2017). "Embedding Words and Senses Together via Joint Knowledge-Enhanced Training". In: *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 100–111.

Manning, Christopher and Hinrich Schutze (1999). *Foundations of statistical natural language processing*. MIT press.

Manning, Christopher D (2015). "Computational linguistics and deep learning". In: *Computational Linguistics* 41.4, pp. 701–707.

Marshall, Max, Shon Lewis, Austin Lockwood, Richard Drake, Peter Jones, and Tim Croudace (2005). "Association between duration of untreated psychosis and outcome in cohorts of first-episode patients: a systematic review". In: *Archives of general psychiatry* 62.9, pp. 975–983.

Maru, Marco, Federico Scozzafava, Federico Martelli, and Roberto Navigli (Nov. 2019). "SyntagNet: Challenging Supervised Word Sense Disambiguation with Lexical-Semantic Combinations". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 3534–3540. DOI: 10.18653/v1/D19-1359. URL: https://aclanthology.org/D19-1359.

McCrae, John P, Ian Wood, and Amanda Hicks (2017). "The colloquial wordnet: Extending princeton wordnet with neologisms". In: *International Conference on Language, Data and Knowledge*. Springer, pp. 194–202.

McCrae, John Philip, Alexandre Rademaker, Ewa Rudnicka, and Francis Bond (2020). "English WordNet 2020: Improving and extending a WordNet for English using an open-source methodology". In: *proceedings of the LREC 2020 workshop on multimodal WordNets (MMW2020)*, pp. 14–19.

Mensa, Enrico, Davide Colla, Marco Dalmasso, Marco Giustini, Carlo Mamo, Alessio Pitidis, and Daniele Paolo Radicioni (2020a). "Violence Detection Explanation Via Semantic Roles Embeddings". In: *BMC Medical Informatics Decis. Mak.* 20.1, p. 263. DOI: 10.1186/s12911-020-01237-4. URL: https://doi.org/10.1186/s12911-020-01237-4.

Mensa, Enrico, Daniele Liberatore, Davide Colla, Matteo Delsanto, Marco Giustini, Daniele Paolo Radicioni, et al. (2022). "Road Accidents: Information Extraction from Clinical Reports." In: *HC@ AIxIA*, pp. 87–97.

Mensa, Enrico, Gian Manuel Marino, Davide Colla, Matteo Delsanto, and Daniele Paolo Radicioni (2020b). "A Resource for Detecting Misspellings and Denoising Medical Text Data". In: *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020, Bologna, Italy, March 1-3, 2021*. Ed. by Johanna Monti, Felice Dell'Orletta, and Fabio Tamburini. Vol. 2769. CEUR Workshop Proceedings. CEUR-WS.org, pp. 1–7. URL: http://ceur-ws.org/Vol-2769/paper_48.pdf.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013a). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013b). "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*, pp. 3111–3119.

Miller, George A (1995). "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11, pp. 39–41.

Miller, George A, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert G Thomas (1994). “Using a semantic concordance for sense identification”. In: *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Moro, Andrea and Roberto Navigli (2015). “Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking”. In: *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pp. 288–297.

Moro, Andrea, Alessandro Raganato, and Roberto Navigli (2014). “Entity linking meets word sense disambiguation: a unified approach”. In: *Transactions of the Association for Computational Linguistics* 2, pp. 231–244.

Mota, Natália B, Mauro Copelli, and Sidarta Ribeiro (2017). “Thought disorder measured as random speech structure classifies negative symptoms and schizophrenia diagnosis 6 months in advance”. In: *npj Schizophrenia* 3.1, pp. 1–10.

Mota, Natalia B, Nivaldo AP Vasconcelos, Nathalia Lemos, Ana C Pieretti, Osame Kinouchi, Guillermo A Cecchi, Mauro Copelli, and Sidarta Ribeiro (2012). “Speech graphs provide a quantitative measure of thought disorder in psychosis”. In: *PloS One* 7.4, e34928.

Nakayama, Hiroki, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang (2018). *doccano: Text Annotation Tool for Human*. Software available from https://github.com/doccano/doccano. URL: `https://github.com/doccano/doccano`.

Náplava, Jakub, Milan Straka, Jana Straková, and Alexandr Rosen (2022). “Czech grammar error correction with a large and diverse corpus”. In: *Transactions of the Association for Computational Linguistics* 10, pp. 452–467.

Navigli, Roberto (2009). “Word sense disambiguation: A survey”. In: *ACM Computing Surveys (CSUR)* 41.2, p. 10.

Navigli, Roberto, David Jurgens, and Daniele Vannella (2013). “Semeval-2013 task 12: Multilingual word sense disambiguation”. In: *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pp. 222–231.

Navigli, Roberto and Federico Martelli (2019). “An overview of word and sense similarity”. In: *Natural Language Engineering* 25.6, pp. 693–714.

Neelakantan, Arvind, Jeevan Shankar, Alexandre Passos, and Andrew McCallum (2014). "Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space". In: *EMNLP*. Ed. by Alessandro Moschitti, Bo Pang, and Walter Daelemans. ACL, pp. 1059–1069. ISBN: 978-1-937284-96-1.

Ng, Hwee Tou, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant (2014). "The CoNLL-2014 shared task on grammatical error correction". In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pp. 1–14.

Ng, Hwee Tou, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault (Aug. 2013). "The CoNLL-2013 Shared Task on Grammatical Error Correction". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 1–12. URL: https://aclanthology.org/W13-3601.

Ngiam, Jiquan, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng (2011). "Multimodal deep learning". In: *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 689–696.

OpenAI, R (2023). "Gpt-4 technical report. arxiv 2303.08774". In: *View in Article* 2, p. 13.

OpenAI, TB (2022). *Chatgpt: Optimizing language models for dialogue. OpenAI.*

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. (2022). "Training language models to follow instructions with human feedback". In: *Advances in neural information processing systems* 35, pp. 27730–27744.

Parra, Camilo, Carlos Ponce, and Salas F Rodrigo (2020). "Evaluating the performance of explainable machine learning models in traffic accidents prediction in california". In: *2020 39th International Conference of the Chilean Computer Science Society (SCCC)*. IEEE, pp. 1–8.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global Vectors for Word Representation." In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing(EMNLP)*. Vol. 14, pp. 1532–1543.

Perez, Luis and Jason Wang (2017). "The effectiveness of data augmentation in image classification using deep learning". In: *arXiv preprint arXiv:1712.04621*.

Peters, Matthew E., Waleed Ammar, Chandra Bhagavatula, and Russell Power (July 2017). "Semi-supervised sequence tagging with bidirectional language models". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 1756–1765. DOI: 10.18653/v1/P17-1161. URL: https://aclanthology.org/P17-1161.

Peters, Matthew E, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih (2018). "Dissecting contextual word embeddings: Architecture and representation". In: *EMNLP, Association for Computational Linguistics*, pp. 1499–1509.

Pianta, Emanuele, Luisa Bentivogli, and Christian Girardi (2002). "MultiWordNet: developing an aligned multilingual database". In: *First international conference on global WordNet*, pp. 293–302.

Pilehvar, Mohammad Taher and Nigel Collier (2016). "De-Conflated Semantic Representations". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1680–1690.

Pilehvar, Mohammad Taher and Roberto Navigli (2015). "From senses to texts: An all-in-one graph-based approach for measuring semantic similarity". In: *Artificial Intelligence* 228, pp. 95–128.

Pitidis, Alessio, Gianni Fondi, Marco Giustini, Eloïse Longo, Giuseppe Balducci, Gruppo di lavoro SINIACA-IDB, and Dipartimento di Ambiente e Connessa Prevenzione Primaria, ISS (2014). "Il Sistema SINIACA-IDB per la sorveglianza degli incidenti". In: *Notiziario dell'Istituto Superiore di Sanità* 27.2, pp. 11–16.

Pradhan, Sameer, Edward Loper, Dmitriy Dligach, and Martha Palmer (2007). "Semeval-2007 task-17: English lexical sample, srl and all words". In: *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)*, pp. 87–92.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever (2019). "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8, p. 9.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu (2019). "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *arXiv preprint arXiv:1910.10683*.

Raganato, Alessandro, Jose Camacho-Collados, and Roberto Navigli (2017). "Word sense disambiguation: A unified evaluation framework and empirical comparison". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 99–110.

Ramshaw, Lance A and Mitchell P Marcus (1999). "Text chunking using transformation-based learning". In: *Natural language processing using very large corpora*. Springer, pp. 157–176.

Rei, Marek (2017). "Semi-supervised multitask learning for sequence labeling". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pp. 2121–2130.

Reisinger, Joseph and Raymond J Mooney (2010). "Multi-prototype vector-space models of word meaning". In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pp. 109–117.

Rudnicka, Ewa Katarzyna, Wojciech Witkowski, and Michał Kaliński (2015). "Towards the methodology for extending princeton wordnet". In: *Cognitive Studies* 15.

Salas, Angelica, Panagiotis Georgakis, C Nwagboso, Ahmad Ammari, and Ionnis Petalas (2017). "Traffic event detection framework using social media". In: *2017 IEEE International Conference on Smart Grid and Smart Cities (ICSGSC)*. IEEE, pp. 303–307.

Salas, Angelica, Panagiotis Georgakis, and Yannis Petalas (2017). "Incident detection using data from social media". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 751–755.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108*.

Santos, Daniel, José Saias, Paulo Quaresma, and Vítor Beires Nogueira (2021). "Machine learning approaches to traffic accident analysis and hotspot prediction". In: *Computers* 10.12, p. 157.

Scarlini, Bianca, Tommaso Pasini, and Roberto Navigli (2020a). "SensEmBERT: Context-Enhanced Sense Embeddings for Multilingual Word Sense Disambiguation". In:

Proceedings of the Thirty-Fourth Conference on Artificial Intelligence. Association for the Advancement of Artificial Intelligence, pp. 8758–8765.

— (2020b). "With More Contexts Comes Better Performance: Contextualized Sense Embeddings for All-Round Word Sense Disambiguation". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Schütze, Hinrich (1998). "Automatic word sense discrimination". In: *Computational linguistics* 24.1, pp. 97–123.

Schütze, Hinrich and Jan O Pedersen (1997). "A cooccurrence-based thesaurus and two applications to information retrieval". In: *Information Processing & Management* 33.3, pp. 307–318.

Scozzafava, Federico, Alessandro Raganato, Andrea Moro, and Roberto Navigli (2015). "Automatic identification and disambiguation of concepts and named entities in the multilingual Wikipedia". In: *Congress of the Italian Association for Artificial Intelligence*. Springer, pp. 357–366.

Selinker, Larry (1972). "Interlanguage". In: *International Review of Applied Linguistics* 10.3, pp. 209–231.

Shen, Yikang, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville (2017). "Neural language modeling by jointly learning syntax and lexicon". In: *arXiv preprint arXiv:1711.02013*.

Shrout, Patrick E and Joseph L Fleiss (1979). "Intraclass correlations: uses in assessing rater reliability." In: *Psychological bulletin* 86.2, p. 420.

Siino, Marco, Elisa Di Nuovo, Ilenia Tinnirello, and Marco La Cascia (2022). "Fake News Spreaders Detection: Sometimes Attention Is Not All You Need". In: *Information* 13.9, p. 426.

Snyder, Benjamin and Martha Palmer (2004). "The English all-words task". In: *Proceedings of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pp. 41–43.

Speer, Robert and Joshua Chin (2016). *An Ensemble Method to Produce High-Quality Word Embeddings*.

Speer, Robyn and Joanna Lowry-Duda (2017). “ConceptNet at SemEval-2017 Task 2: Extending Word Embeddings with Multilingual Relational Knowledge”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 85–89. DOI: 10.18653/v1/S17-2008. URL: http://aclweb.org/anthology/S17-2008.

Stahlberg, Felix and Shankar Kumar (2021). “Synthetic data generation for grammatical error correction with tagged corruption models”. In: *arXiv preprint arXiv:2105.13318*.

Stolcke, Andreas and Elizabeth Shriberg (1996). “Statistical language modeling for speech disfluencies”. In: *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*. Vol. 1. IEEE, pp. 405–408.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). “Sequence to sequence learning with neural networks”. In: *Advances in neural information processing systems*, pp. 3104–3112.

Tenney, Ian, Dipanjan Das, and Ellie Pavlick (July 2019). “BERT Rediscovers the Classical NLP Pipeline”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 4593–4601. DOI: 10.18653/v1/P19-1452. URL: https://aclanthology.org/P19-1452.

Tenney, Ian, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, and Ellie Pavlick (2019). “What do you learn from context? probing for sentence structure in contextualized word representations”. In: *arXiv preprint arXiv:1905.06316*.

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. (2023). “Llama 2: Open foundation and fine-tuned chat models”. In: *arXiv preprint arXiv:2307.09288*.

Treccani (n.d.). *Treccani Italian Dictionary*. URL: \url{https://treccani.it}.

UK Department for Transport (Accessed: 10-10-2022). *Road Safety Data*. https://www.data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data.

UNECE (n.d.). *Definitions of Vehicles (pages 5-11)*. URL: \url{https://unece.org/fileadmin/DAM/trans/main/wp29/wp29resolutions/ECE-TRANS-WP.29-78r6e.pdf}.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*, pp. 5998–6008.

Vigorelli, Pietro (2004). *La conversazione possibile con il malato Alzheimer*. Vol. 48. FrancoAngeli.

— (2010). "The ABC Group for caregivers of persons living with dementia: self-help based on the Conversational and Enabling Approach". In: *Non-pharmacological therapies in dementia* 3, pp. 271–286.

— (2011). "L'approccio capacitante: come prendersi cura degli anziani fragili e delle persone malate di Alzheimer". In.

— (2021). "Dialoghi imperfetti. Per una comunicazione felice nella vita quotidiana e nel mondo Alzheimer." In.

Vigorelli, Pietro Enzo, Nicola Benvenuti, and Andrea Bolioli (1994). "Il Corpus Anchise, un insieme di 320 conversazioni con persone con demenza". In: *Neurology* 44, pp. 2308–2314.

Vilnis, Luke and Andrew McCallum (2014). "Word representations via gaussian embedding". In: *arXiv preprint arXiv:1412.6623*.

Volodina, Elena, Christopher Bryant, Andrew Caines, Orphée De Clercq, Jennifer-Carmen Frey, Elizaveta Ershova, Alexandr Rosen, and Olga Vinogradova (2023). "MultiGED-2023 shared task at NLP4CALL: Multilingual Grammatical Error Detection". In: *Proceedings of the 12th workshop on Natural Language Processing for Computer Assisted Language Learning (NLP4CALL)*. Tórshavn, Faroe Islands, pp. 1–15.

Volodina, Elena, Lena Granstedt, Arild Matsson, Beáta Megyesi, Ildikó Pilán, Julia Prentice, Dan Rosén, Lisa Rudebeck, Carl-Johan Schenström, Gunlög Sundberg, et al. (2019). "The swell language learner corpus: From design to annotation". In: *Northern European Journal of Language Technology (NEJLT)* 6, pp. 67–104.

Vu, Thuy and D Stott Parker (2016). "K-Embeddings: Learning Conceptual Embeddings for Words using Context". In: *Proceedings of the 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 1262–1267.

Walenski, Matthew, Thomas W Weickert, Christopher J Maloof, and Michael T Ullman (2010). "Grammatical processing in schizophrenia: Evidence from morphology". In: *Neuropsychologia* 48.1, pp. 262–269.

Wang, Alex, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman (2019). "Superglue: A stickier benchmark for general-purpose language understanding systems". In: *arXiv preprint arXiv:1905.00537*.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman (2018). "GLUE: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461*.

White, Jules, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt (2023). "A prompt pattern catalog to enhance prompt engineering with chatgpt". In: *arXiv preprint arXiv:2302.11382*.

Wolf, Thomas et al. (Oct. 2020). "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, and Klaus Macherey (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144*.

Yaghoobzadeh, Yadollah and Hinrich Schütze (2016). "Intrinsic subspace evaluation of word embedding representations". In: *arXiv preprint arXiv:1606.07902*.

Yang, Xi, Jiang Bian, William R Hogan, and Yonghui Wu (Oct. 2020). "Clinical concept extraction using transformers". In: *Journal of the American Medical Informatics Association*. ocaa189. ISSN: 1527-974X. DOI: 10.1093/jamia/ocaa189. eprint: https://academic.oup.com/jamia/advance-article-pdf/doi/10.1093/

jamia/ocaa189/34055422/ocaa189.pdf. URL: https://doi.org/10.1093/jamia/ocaa189.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). “XLNet: Generalized Autoregressive Pretraining for Language Understanding”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc.

Yannakoudakis, Helen, Ted Briscoe, and Ben Medlock (June 2011). “A New Dataset and Method for Automatically Grading ESOL Texts”. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, pp. 180–189. URL: https://aclanthology.org/P11-1019.

Yannakoudakis, Helen, Øistein E Andersen, Ardeshir Geranpayeh, Ted Briscoe, and Diane Nicholls (2018). “Developing an automated writing placement system for ESL learners”. In: *Applied Measurement in Education* 31.3, pp. 251–267.

Yu, Dong, Yifan Gong, Michael Alan Picheny, Bhuvana Ramabhadran, Dilek Hakkani-Tür, Rohit Prasad, Heiga Zen, Jan Skoglund, Jan Honza Černockỳ, Lukáš Burget, et al. (2023). “Twenty-Five Years of Evolution in Speech and Language Processing”. In: *IEEE Signal Processing Magazine* 40.5, pp. 27–39.

Yuan, Zheng, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant (Nov. 2021). “Multi-Class Grammatical Error Detection for Correction: A Tale of Two Systems”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 8722–8736. DOI: 10.18653/v1/2021.emnlp-main.687. URL: https://aclanthology.org/2021.emnlp-main.687.

Zhang, Zheyuan, Jifan Yu, Juanzi Li, and Lei Hou (2023). “Exploring the Cognitive Knowledge Structure of Large Language Models: An Educational Diagnostic Assessment Approach”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 1643–1650.

Zhou, Yongchao, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba (2022). “Large language models are human-level prompt engineers”. In: *arXiv preprint arXiv:2211.01910*.

Ziegler, Daniel M, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving (2019). "Fine-tuning language models from human preferences". In: *arXiv preprint arXiv:1909.08593*.

Économiques, Institut National de la Statistique et des Études (n.d.). *Definitions: Road Accidents*. URL: \url{https://www.insee.fr/en/metadonnees/definition/c1116}.