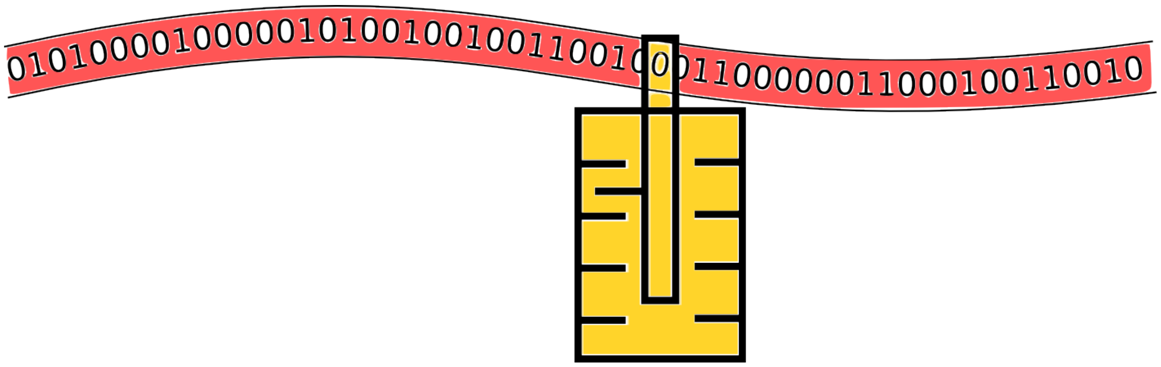


Matteo Baldoni, Federico Chesani,
Bernardo Magnini, Paola Mello, Marco Montali (eds.)

Popularize Artificial Intelligence

*AI*IA Workshop and Prize*
for celebrating 100th anniversary of
Alan Turing's birth, PAI 2012

Rome, Italy, June 15th, 2012

Workshop Notes

PAI 2012 Home Page:
`http://www.di.unito.it/~baldoni/PAI-2012/`

Preface

In 1934 *Alan M. Turing* argued that machines could imitate thought and, in the essay "*Computing Machinery and Intelligence*", he proposed what is now known as the *Turing test.* A computer and a human are compared by a panel of judges, who address the same questions to both and review their answers. If the judges cannot make distinctions between the two answers, the machine may be considered intelligent.

The Turing test for decades has represented the best calling card for popularizing Artificial Intelligence worldwide. Nowadays, ideas, theories, techniques developed in the research area of Artificial Intelligence are widely applied for producing games, satellite navigators, human-computer interfaces, web applications, automated voice responders, etc.

For celebrating the *100th anniversary of Alan Turing's birth*, the *Italian Association for Artificial Intelligence* (AI*IA) organizes the workshop *Popularize Artificial Intelligence* (PAI), which is aimed at divulging the practical uses of Artificial Intelligence. The workshop PAI 2012 is organized in the context of the *12th AI*IA Symposium on Artificial Intelligence.*

The papers included in these proceedings present demonstrators, demos, proposals, systems exploiting AI theories and techniques, which are written in such a way to make them accessible to a broad public. In particular, they are organized in three categories of contribution: (1) *industrial experiences*, (2) *research and academic experiences*, (3) *student experiences inside AI courses.*

This edition of PAI received *seventeen* submissions, *sixteen* of which have been selected by the Programme Committee and are included in this volume. Each paper received at least three reviews in order to supply the authors with helpful feedback. In particular, the Programme Committee and the Workshop Organisers have selected the following papers, one for each category, as the best papers:

- Industrial experiences: "*Miglioramento di algoritmi di elaborazione di immagini da scanner 3D tramite Simulated Annealing*", by Marco Derboni, Evelina Lamma and Antonio Zaccaro;
- Research and academic experiences: "*TrentinoMedia: Exploiting NLP and Background Knowledge to Browse a Large Multimedia News Store*", by Roldano Cattoni, Francesco Corcoglioniti, Christian Girardi, Bernardo Magnini, Luciano Serafini and Roberto Zanoli;
- Student experiences inside AI courses: "*DALI Logical Agents into Play*", by Stefania Costantini, Niva Florio, Giovanni De Gasperis, Annalisa D'Andrea and Arianna Tocchio.

We would like to thank all authors for their contributions and the members of the Programme Committee for their excellent work during the reviewing phase. Finally, we would like to thank also the organizers of the 12th AI*IA Symposium on Artificial Intelligence for hosting the workshop and the Italian Association for Artificial Intelligence for providing the prizes and the grants for the authors of the best papers and students.

June 6th, 2012

Matteo Baldoni
Federico Chesani
Bernardo Magnini
Paola Mello
Marco Montali

Workshop Organisers

Matteo Baldoni	University of Torino, Italy
Federico Chesani	University of Bologna, Italy
Bernardo Magnini	Fondazione Bruno Kessler, Italy
Paola Mello	University of Bologna, Italy
Marco Montali	University of Bozen-Bolzano, Italy

Programme Committee

Giuliano Armano	University of Cagliari, Italy
Roberto Basili	University of Roma Tor Vergata, Italy
Federico Bergenti	University of Parma, Italy
Luigia Carlucci Aiello	University of Rome "La Sapienza", Italy
Federica Cena	University of Torino, Italy
Stefania Costantini	University of L'Aquila, Italy
G. Barbara Demo	University of Torino, Italy
Nicola Di Mauro	University of Bari, Italy
Stefano Ferilli	University of Bari, Italy
Paolo Frasconi	University of Firenze, Italy
Maurizio Gabbrielli	University of Bologna, Italy
Rosella Gennari	University of Bozen-Bolzano, Italy
Marco Gori	University of Siena, Italy
Nicola Leone	University of Cosenza, Italy
Leonardo Lesmo	University of Torino, Italy
Viviana Mascardi	University of Genova, Italy
Emanuele Menegatti	University of Padova, Italy
Daniele Nardi	University of Rome "La Sapienza", Italy
Andrea Omicini	University of Bologna, Italy
Roberto Pirrone	University of Palermo, Italy
Piero Poccianti	MPS, Italy
Agostino Poggi	University of Parma, Italy
Andrea Roli	University of Bologna, Italy
Gianfranco Rossi	University of Parma, Italy
Marco Schaerf	University of Rome "La Sapienza", Italy
Giovanni Semeraro	University of Bari, Italy

Additional Reviewers

Baroglio, Cristina	Leuzzi, Fabio	Ricca, Francesco
Calimeri, Francesco	Mauro, Jacopo	Schifanella, Claudio
Caputo, Annalina	Narducci, Fedelucio	

Copyright

Sponsoring Institutions

Associazione Italiana per l'Intelligenza Artificiale

Table of Contents

Industrial Experiences

Un sistema di Vision Inspection basato su reti neurali 1
Ludovico Buffon, Evelina Lamma, Fabrizio Riguzzi, Davide Forment

Miglioramento di algoritmi di elaborazione di immagini da scanner 3D tramite Simulated Annealing 7
Marco Derboni, Evelina Lamma, Antonio Zaccaro

I2AM: a Semi-Automatic System for Data Interpretation in Petroleum Geology 14
Denis Ferraretti, Giacomo Gamberoni, Evelina Lamma

Research and Academic Experiences

TrentinoMedia: Exploiting NLP and Background Knowledge to Browse a Large Multimedia News Store 21
Roldano Cattoni, Francesco Corcoglioniti, Christian Girardi, Bernardo Magnini, Luciano Serafini, Roberto Zanoli

Multimodal interaction with emotional feedback 27
Francesco Cutugno, Antonio Origlia, Roberto Rinaldi

Human action recognition oriented to humanoid robots action reproduction 35
Stefano Michieletto, Emanuele Menegatti

Semantics and Inference for Probabilistic Ontologie 41
Fabrizio Riguzzi, Evelina Lamma, Elena Bellodi, Riccardo Zese

OTTHO: An Artificial Player for a Complex Language Game 47
Giovanni Semeraro, Pasquale Lops, Marco de Gemmis, Pierpaolo Basile

Towards an Interactive Personal Care Sytem driven by Sensor Data 54
Stefano Bragaglia, Paola Mello, Davide Sottara

Student Experiences Inside AI Courses

4D-Particle filter localization for a simulated UAV 60
Anna-Chiara Bellini

Behind the scenes of Sudoku — Application of genetic algorithms for the optimal fun 67
Thomas Bridi

ProgramD_TV! 72
Emiliano Caponi, Irene Fabrini, Andrea Vanzo

DALI Logical Agents into Play 79
Stefania Costantini, Niva Florio, Giovanni De Gasperis, Annalisa D'Andrea, Arianna Tocchio

Artificial Intelligence applied on the Risk! game 86
Luca Fuligni, Andrea Franzon, Orfeo Ciano

A Neural Network for Automatic Vehicles Guidance 92
Alessandro Ghidotti Piovan

An Artificial Intelligence that plays for competitive Scrabble 98
Alberto Strade, Fulvio Di Maria

Author Index 104

Un sistema di Vision Inspection basato su reti neurali

Ludovico Buffon[1], Evelina Lamma[1], Fabrizio Riguzzi[1], Davide Formenti[2]

[1]Dipartimento di Ingegneria, Via Saragat 1, 44122 Ferrara (FE), Italia
ludovico.buffon@student.unife.it,
{evelina.lamma, fabrizio.riguzzi}@unife.it
[2]Bonfiglioli Engineering, Vigarano Pieve (FE), Italia
d.formenti@bonfiglioliengineering.com

Abstract. Il lavoro descrive un sistema di Intelligenza Artificiale per estrarre e riconoscere difetti presenti su flaconi per uso cosmetico e farmaceutico. Il sistema realizzato, nel suo complesso, integra algoritmi di *corner detection* e *clustering* per l'identificazione delle regioni di interesse che costituiscono potenziali difetti, e algoritmi di classificazione automatica basati su reti neurali. Bonfiglioli Engineering s.p.a. ha fornito un database di oltre 20.000 immagini (di flaconi con difetti o integri) a cui fare riferimento, sia per la fase di addestramento sia per quella di *testing* della rete neurale.

Keywords. Visione automatica, Classificazione automatica, Reti neurali.

1 Introduzione

La visione automatica (*Computer Vision*) ha lo scopo di ricavare informazioni qualitativamente elevate con l'obiettivo di comprendere l'immagine per individuare, ad esempio, qualche *pattern* in essa contenuto. In campo industriale, la visione automatica ricopre ormai un ruolo di primo piano in diverse applicazioni di misura e controllo, grazie alla disponibilità di potenza di calcolo adeguata all'esecuzione di algoritmi di visione e classificazione automatica *real-time* sempre più complessi. I problemi industriali affrontabili mediante visione automatica sono eterogenei [3].

In questo lavoro, descriviamo un sistema software basato su tecniche di Intelligenza Artificiale e di apprendimento automatico per il riconoscimento di *pattern* da immagini ([1,3]), in particolare difetti in flaconi per uso cosmetico e farmaceutico. I principali difetti da ricercare sono bolle d'aria nel vetro, crepe, punti neri, macchie e impurità del materiale, graffi estesi a tutta la superficie (denominati *scratch*). In Figura 1, a sinistra, sono riportati alcuni esempi di difetti. I difetti sono da ricercare su tutto il flacone, in particolare (si veda Fig.1, a destra), nella base, nella parte cilindrica, nella spalla, nel collo del flacone. Il caso di studio è stato proposto da Bonfiglioli Engineering s.p.a. per il miglioramento dei propri prodotti, fornendo anche un database di oltre 20.000 immagini, molto eterogeneo e comprendente immagini di varia grandezza e risoluzione, in cui ogni flacone (difettoso o meno) è fotografato da varie angolazioni.

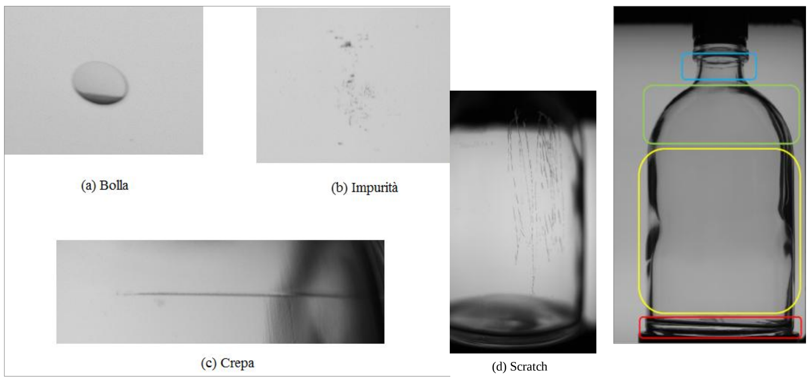


Fig. 1. Esempi di difetti, a dx e centro; Parti del flacone (base, cilindro, spalla, collo), a sx

Nell'affrontare il problema, si è preferito un approccio basato su reti neurali rispetto ad altri, perché: *(i)* è già stato applicato con successo in campi simili (ad esempio, per riconoscimento di volti), *(ii)* garantisce, inoltre, la capacità della rete di addestrarsi da esempi adattandosi ai *pattern* in ingresso, con un algoritmo efficiente di apprendimento dei pesi della rete (*back-propagation*), e *(iii)* garantisce la capacità di generalizzazione per trattare *input* mai visti in precedenza. Tuttavia, questo approccio ha presentato inizialmente un ostacolo, perché, per efficienza computazionale, esso è in grado di trattare *pattern* di ingresso di estensione limitata. Per questo motivo, il sistema realizzato integra, nel suo complesso, algoritmi di classificazione automatica basati su reti neurali ([2,6]) e, a monte di questi, applica algoritmi di *corner detection* [7] e *clustering* per identificare nell'immagine (molto estesa in termini di *pixel*) regioni di interesse (*Region Of Interest*, ROI) di area più limitata e le loro caratteristiche, che costituiscono potenziali difetti, successivamente classificate dalla rete neurale addestrata.

2 L'architettura software

Il software è stato realizzato in C# con l'ausilio di alcune librerie per il trattamento di immmagini `.tiff` e altre per l'apprendimento automatico. In particolare, nello sviluppo, sono state utilizzate la libreria `Accord.net` [4] e la libreria `OpenCvSharp`, porting su piattoforma `.Net` della omonima libreria Intel per visione artificiale. L'interfaccia utente principale comprende ausili per eseguire sia la fase di addestramento della rete, sia la successiva fase di *testing* e di applicazione del classificatore appreso. La topologia della rete neurale (Multi-Layered Perceptron, MLP [8]) ha uno strato di *input*, uno di elaborazione ed uno di *output*. La fase di training avviene sfruttando un algoritmo di *error back-propagation*.

Nella fase di *Training*, si presentano delle immagini all'operatore per allenare la rete neurale al riconoscimento dei *pattern* stabiliti. La fase di addestramento, quindi, è quella che richiede il maggior tempo d'intervento dell'esperto per individuare gli esempi di *training* più idonei alle circostanze e agli obiettivi dello studio. La scelta delle immagini di *training* deve essere accurata perché, tramite le informazioni in esse contenute, saranno estratte le firme spettrali delle varie classi. Gli *input* della rete sono costituiti da aree estratte dall'esperto dalle fotografie, dette *Region Of Interest* (ROI). Nella fase di *training*, le ROI sono aree di dimensione 30x30 pixel, identificate da parte dell'esperto. Alla rete neurale è data in *input* l'intera ROI effettuando una mappatura uno ad uno fra pixel e neuroni di *input*, insieme alla classe attribuita dall'esperto (si veda Figura 2).

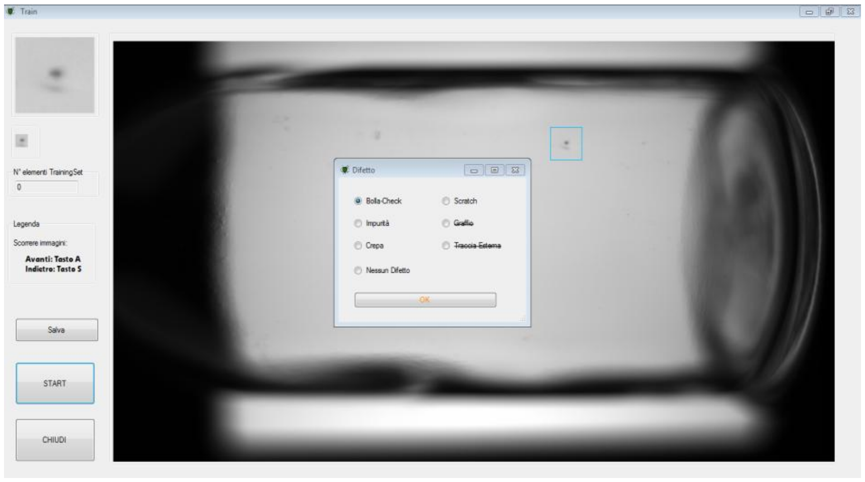


Fig. 2. Fase di *training*

Dopo aver opportunamente allenato la rete a riconoscere i difetti proposti, essa può essere utilizzata per una classificazione automatica degli stessi. Il sistema si compone di due passi: la prima di estrazione automatica delle caratteristiche, la seconda di classificazione automatica. Lo schema complessivo è riportato in Figura 3.

La fase di *feature extraction* si divide, a sua volta, in due passi in cascata. Nel primo, si analizza l'immagine con l'algoritmo di *corner detection* FAST (Features from Accelerated Segment Test [7]) per selezionarne i punti caratteristici (o *key point*), salvati poi in un vettore di punti in coordinate cartesiane. Successivamente questi punti sono raggruppati in *cluster* tramite l'algoritmo DBScan (Density-Based Spatial Clustering of Applications with Noise [5]). Infatti i punti che l'algoritmo FAST individua si addensano in prossimità dei difetti presenti nell'immagine e quindi i *cluster*, opportunamente filtrati per mantenere quelli con più punti al loro interno, individuano regioni spaziali dove c'è più probabilità di avere un difetto. Dopo aver selezionato i *cluster* di interesse, si estrae una regione di 30x30 pixel (Region Of Interest, ROI), passata poi alla rete neurale (MLP, in Figura 3) per essere classificata. La classe individuata è quella per cui la probabilità calcolata dalla rete è massima [9].

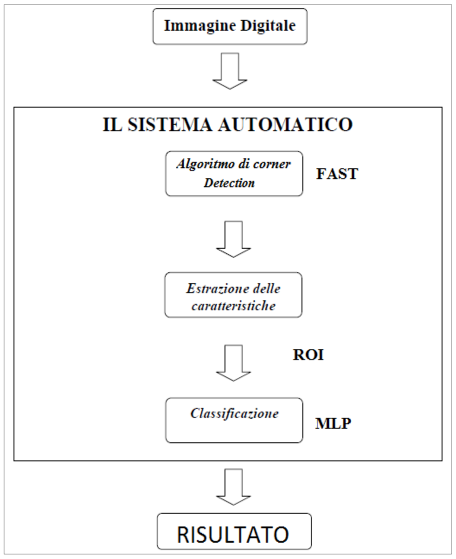


Fig. 3. Schema del sistema di classificazione automatica

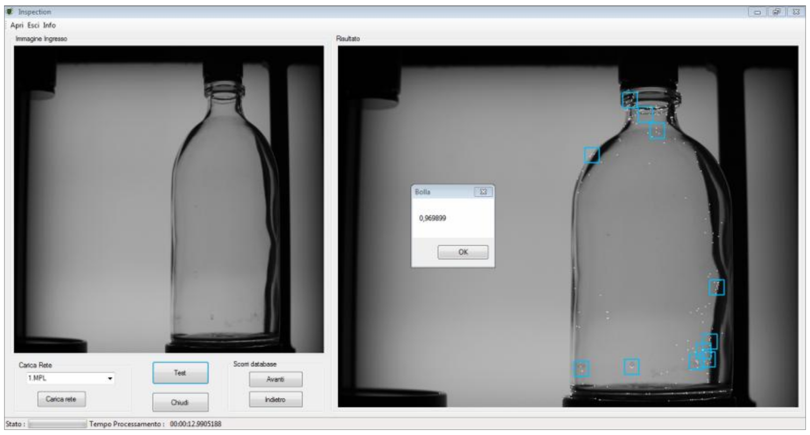
3 Sperimentazione e risultati

Il database messo a disposizione contava più di 20.000 immagini, comprensivo sia di immagini di flaconi con difetti sia non difettosi, fotografati ciascuno da quattro diverse angolazioni. La fase di addestramento e la successiva di test sono state effettuato su un numero ridotto di immagini, circa 12.000, partizionate in due insiemi, rispettivamente per il *Training* (7000 immagini) e il *Test* (5000 immagini) della rete.

Dal punto di vista hardware la fase di *Test*, che è quella che richiede la maggior parte delle risorse della macchina su cui è eseguito il programma, occupa al massimo un quantitativo di memoria pari a 250 Mb ed esegue l'elaborazione e la classificazione di un'immagine in un tempo non superiore ai 2 secondi.

Nella fase di *Test* si è valutata inizialmente la bontà del sistema proposto analizzando le immagini con sole due possibili classi di appartenenza: `Difetto/Non-Difetto`. In questa prima sperimentazione svolta, si è valutata la capacità del sistema sviluppato addestrando la rete con 200 esempi selezionati dal *dataset* di 7.000 immagini. La scelta di usare solo 200 immagini nel *training* è dovuto al fatto che questa fase è risultata molto lunga e dispendiosa in termini di tempo uomo. Il database di *test* conteneva invece le restanti circa 5.000 immagini. In questa fase circa il 97% delle immagini (4850 immagini su 5000) è stato classificato correttamente.

In seguito, si è aumentato il numero di classi per avere una classificazione più precisa. Nell'attuale sviluppo del sistema, il numero di classi che la rete neurale riesce a discriminare è pari a cinque: `Non-Difetto`, `Bolla`, `Crepa`, `Impurità`, più la classe denominata `Scratch`. Con l'aumento del numero di classi, nella seconda sperimentazione condotta, per mantenere una buona classificazione, si è dovuto aumentare sia il numero di esempi forniti alla rete (portato a 320 esempi) sia il numero di cicli di apprendimento, e diminuire il *learning rate* per evitare ampie oscillazioni dell'errore medio sul *training set.* I tempi di apprendimento si sono allungati fino a qualche ora con 320 esempi. È stato poi preso in considerazione per il *test* solo un sottoinsieme di 500 immagini prelevate a caso dal database di 5.000 totali per un ammontare di circa 1.000 difetti estratti e analizzati. I risultati evidenziano che è rimasta circa invariata la capacità di *detection* delle problematiche presenti dovuto al fatto che FAST è un algoritmo con un'ottima stabilità e ripetibilità nell'individuazione dei punti interessanti. La classificazione più accurata ha evidenziato un errore del 7-8% nella corretta classificazione della regione estratta. All'interno delle 5 classi la probabilità maggiore di errore è data dalla discriminazione tra `Crepa` e `Impurità` in regioni con poco contenuto informativo, mentre la classificazione come `Bolla` ha dato il 98% di accuratezza. Un esempio di elaborazione è mostrato in Figura 4, con l'immagine di ingresso a sinistra, e i *cluster* interessanti e classificati evidenziati direttamente sull'immagine a destra per un immediato riscontro visivo.



Bolla

0,969899

1.MPL

Stato : Tempo Processamento : 00:00:12.9905188

Fig. 4. Classificazione dei *cluster* interessanti

4 Conclusioni e sviluppi futuri

In questo lavoro abbiamo integrato reti neurali e algoritmi di *corner detection* e *clustering* automatico per l'estrazione delle caratteristiche visuali da immagini in un sistema software per l'identificazione di difetti in flaconi ad uso cosmetico e farmaceutico, prodotti dai macchinari sviluppati da Bonfiglioli Engineering s.p.a. Il sistema realizzato è, per ora, funzionante *off-line* rispetto alla linea di produzione dell'azienda. Il sistema, dopo la fase di addestramento della rete neurale, ha evidenziato un'ottima capacità di individuare i difetti presenti all'interno delle immagini e una buona capacità di classificare il *pattern* estratto.

Il processo di ricerca dei parametri ottimali per l'algoritmo di classificazione automatica realizzato dalla rete neurale è stato lo scoglio maggiore del progetto. Sono stati provati diversi valori di inizializzazione in modo tale da avvicinarsi ad un buon comportamento della rete, e ottenendo buoni risultati sperimentali.

Grazie alla modularità del software realizzato, si possono facilmente inserire ulteriori difetti, rispetto a quelli già considerati, già presenti nel database di immagini fornito dall'azienda, come `TracceSottili`, `Graffi`, e `TracceEsterne`.

Ulteriori sviluppi futuri riguardano l'integrazione di nuovi classificatori come macchine SVM, Hidden Markov Model, AdaBoost, Reti di Kohonen (SOM), già previsti nella struttura del codice e parzialmente considerati in un lavoro parallelo a questo come approcci alternativi.

Ringraziamenti

Il lavoro è stato sviluppato nell'ambito del progetto dal titolo *Elaborazione delle Immagini e Visione Artificiale per la Classificazione di Immagini in ambito Industriale*, finanziato con il contributo della Camera di Commercio, Industria, Artigianato e Agricoltura di Ferrara, Bando CCIAA-2010.

Riferimenti bibliografici

1. Bishop, C.M., "Pattern pecognition and machine learning". Springer, 2006.
2. Bishop, C. M., "Neural Networks for Pattern Recognition". Oxford University Press, 1995.
3. Cucchiara R., Mello P., Piccardi M., Riguzzi F., "An application of machine learning and statistics to defect detection", Intelligent Data Analysis, 5(2):151-164, March/April 2001.
4. `http://accord-net.origo.ethz.ch`
5. `http://it.wikipedia.org/wiki/Dbscan`
6. Leoneds, C.T, (editor), " Image processing and pattern recognition - Neural network system techniques and applications", Academic Press, 1998.
7. Rosten, E., Drummond, T., "Machine learning for high-speed corner detection", Proc. European Conference on Computer Vision, vol. 1, pp. 430-443, 2006.
8. Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain". Cornell Aeronautical Laboratory, Psychological Review}, 65:386, 1958.
9. Vapnik, V.N., "The Nature of Statistical Learning Theory", second edition, Springer, 1999.

Miglioramento di algoritmi di elaborazione di immagini da scanner 3D tramite Simulated Annealing

Marco Derboni[1], Evelina Lamma[1], Antonio Zaccaro[2]

[1]Dipartimento di Ingegneria, Via Saragat 1, 44122 Ferrara (FE), Italia
marco.derboni@student.unife.it, m.derboni@gmail.com,
evelina.lamma@unife.it
[2]Cefla Dental Group, via Bicocca 14/c, 40026 Imola (BO), Italia
antonio.zaccaro@cefla.it

Abstract. Nel lavoro si descrive l'applicazione di tecniche di IA per l'elaborazione e l'allineamento di immagini provenienti da scanner 3D, in grado di generare modelli digitali tridimensionali. Al fine di allineare più correttamente i fotogrammi acquisiti, in scansioni successive, dal dispositivo e ricostruire accuratamente il modello tridimensionale digitale, si è integrata la tecnica di Simulated Annealing, con un algoritmo di elaborazione di immagini (Iterative Closest Point, ICP) che allinea un nuovo fotogramma al modello già generato tramite roto-traslazioni consecutive.

Keywords. Ricerca euristica, Simulated Annealing, scanner 3D, registrazione, allineamento.

1 Introduzione

Nel corso degli anni, la necessità di poter manipolare e/o modificare un oggetto senza doverne alterare il reale stato fisico, oppure l'esigenza di volerne preservare le caratteristiche nel tempo in una forma imperturbabile, ha portato allo sviluppo di strumenti di scansione 3D sempre più sofisticati e complessi in grado di acquisire le informazioni relative alle geometrie, alle dimensioni, alle colorazioni di un oggetto sotto forma di modello digitalizzato tramite la tecnica del laser scanning. I sensori di acquisizione consentono di ottenere delle nuvole di punti che necessitano di un'accurata elaborazione.

L'obiettivo del lavoro presentato consiste nell'ottimizzare il processo di elaborazione di queste informazioni al fine di ricostruire, in modo migliorato, il modello tridimensionale digitalizzato dell'oggetto scansionato. Il caso di studio è stato proposto da Cefla Dental Group, ed è stato affrontato utilizzando tecniche di Intelligenza Artificiale.

Il dispositivo di acquisizione considerato è uno scanner intra-orale per la rilevazione dell'impronta digitale delle arcate dentali, da impiegarsi nel settore odontoiatrico, composto da una sonda, manovrata manualmente dall'operatore, e da un software di controllo e acquisizione. Una volta introdotto nel cavo orale, lo scanner 3D effettua la misurazione delle arcate dentali generando un modello tridimensionale topo-morfologico del tutto analogo alla tradizionale impronta in silicone; il modello generato è utilizzato poi per realizzare un preciso manufatto protesico.

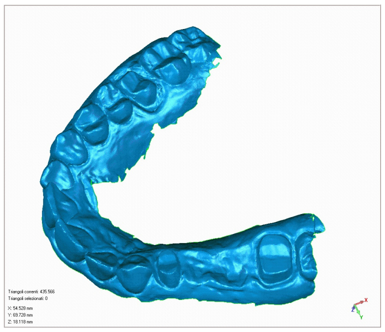


Fig. 1. Modello digitale 3D di un'arcata dentale

Per ottenere un modello corretto e preciso sono, in genere, necessarie più scansioni, ciascuna delle quali produce un'immagine tridimensionale parziale dell'oggetto stesso, quindi un dato volumetrico. I singoli fotogrammi devono essere elaborati attraverso opportuni algoritmi di elaborazione, per produrre un singolo modello tridimensionale dell'arcata scansionata. L'algoritmo di allineamento già utilizzato in Cefla Dental Group, l'Iterative Closest Point (ICP), ha il compito di confrontare la nuvola di punti del fotogramma in esame con quella del modello fino a quel momento generato; se esso individua una regione di sovrapposizione sufficiente riesce ad allineare il fotogramma al modello, in caso contrario attende un nuovo fotogramma e ripete il confronto. Quest'ultima situazione genera una perdita di continuità, cioè una momentanea interruzione del processo di creazione del modello che, conseguentemente, comporta una sospensione temporanea del processo di acquisizione. Se si desidera continuare la scansione, è necessario inquadrare l'ultima area correttamente registrata al modello stesso, in modo da consentire al sistema di riagganciare le scansioni.

Al fine di migliorare l'usabilità del dispositivo, si è pensato di far fronte al problema attraverso una procedura automatica che integra la tecnica del Simulated Annealing con l'algoritmo ICP, minimizzando le situazioni di imprecisione e migliorandone la *performance*, anche temporale.

Nell'articolo si descrive lo studio, la progettazione, la realizzazione e il testing di questa nuova metodologia di allineamento dei fotogrammi acquisiti da scanner tridimensionali, al fine di perfezionare, in termini di accuratezza e di qualità generale, il processo con cui vengono collezionati al fine di consentire l'ottenimento del modello tridimensionale della cavità orale.

2 Integrazione di Simulated Annealing e ICP

L'algoritmo Simulated Annealing [1] è un criterio basato su nozioni di meccanica statistica attraverso un'analogia con il comportamento di sistemi fisici durante il processo di raffreddamento. Ciò che ha influenzato l'implementazione di tale algoritmo è il processo fisico di formazione dei cristalli. Per realizzare un cristallo si parte da materiali grezzi allo stato fuso, la cui temperatura deve essere adeguatamente ridotta fino al momento in cui avviene un congelamento della struttura del cristallo stesso. Il raffreddamento deve avvenire molto lentamente per evitare la formazione di irregolarità al suo interno, quindi la temperatura deve essere abbassata in modo graduale attraverso una serie di livelli. Finché essa è maggiore dello zero assoluto sono sempre possibili variazioni in salita, al fine di evitare che la temperatura si discosti da quella compatibile con il livello energetico del livello corrente. Nel momento in cui viene raggiunto lo zero assoluto, tali variazioni diventano proibite, evitando in questo modo il generarsi di effetti indesiderati nella struttura del cristallo, in quanto nessuna transizione di stato può portare ad uno stato a più alta energia.

Il Simulated Annealing è la versione algoritmica di questo processo di solidificazione dei cristalli. Esso può essere visto come un'estensione della tecnica di ottimizzazione locale, in cui la soluzione iniziale è ripetutamente modificata attraverso piccole perturbazioni locali al fine di migliorarne lo stato. Questo processo mira a trovare un minimo globale quando si è in presenza di più minimi locali.

L'algoritmo Iterative Closest Point (ICP) ha il compito di allineare un fotogramma al modello tramite roto-traslazioni consecutive. L'implementazione utilizzata è stata ideata da Y. Chen e G. Medioni [2]. L'operazione è effettuata a partire da un preciso *starting point*, corrispondente alla regione di sovrapposizione dell'ultimo fotogramma correttamente registrato al modello, ed ha esito positivo solamente se la distanza tra il fotogramma ed il modello, espressa da un'opportuna funzione errore, è inferiore ad una determinata soglia. L'algoritmo associa delle coppie di punti, appartenenti rispettivamente al fotogramma e al modello, in base a dei criteri di distanza. La funzione errore è espressa dalla sommatoria delle distanze di queste coppie di punti. Ad ogni rototraslazione il valore della sommatoria tenderà a diminuire a causa dell'avvicinamento del fotogramma al modello. Questo metodo raggiunge molto rapidamente un minimo, ma non sempre è quello globale, e quindi non corrisponde alla zona di corretto allineamento del fotogramma. Se uno di questi minimi locali è maggiormente adiacente allo *starting point* della scansione corrente rispetto al minimo globale, può accadere che le roto-traslazioni introdotte dall'ICP trascinino la nuvola di punti appartenente alla scansione stessa in una di queste zone.

Simulated Annealing è in grado di sfuggire ai minimi locali, quindi è un algoritmo robusto, tuttavia estremamente lento a convergere al minimo globale, quindi inefficiente. Al contrario, ICP è molto efficiente, ma trova ostacoli nei minimi locali. Da queste considerazioni, nasce l'idea di sviluppare una tecnica ibrida [3] in cui vengono accostati i due metodi al fine di trovare una soluzione ottima al problema.

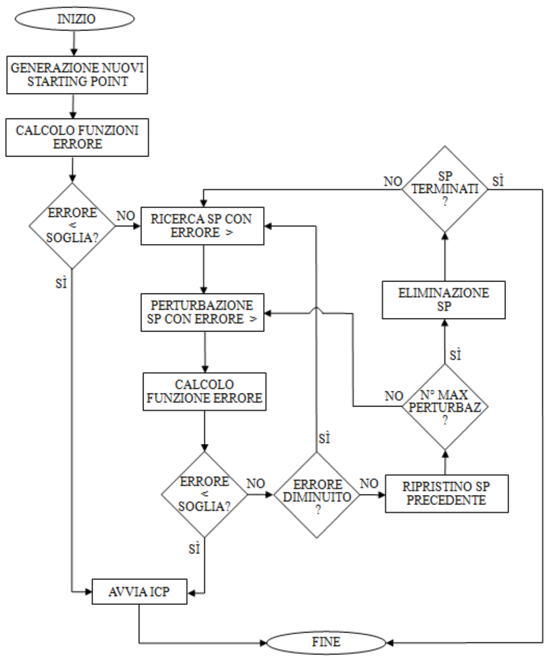


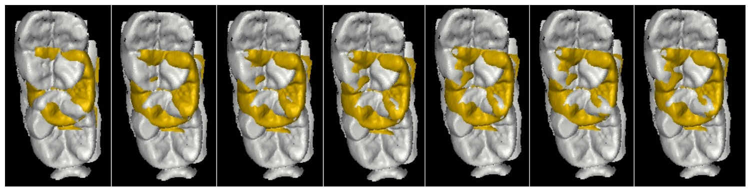
Fig. 2. Schema a blocchi dell'algoritmo Simulated Annealing

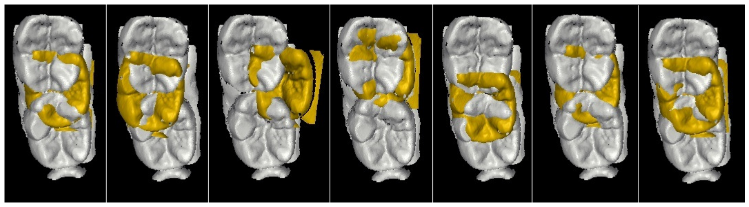
L'algoritmo Simulated Annealing ha infatti il compito di trovare uno *starting point* migliore di quello utilizzato da ICP, per portare a termine la registrazione della nuvola di punti acquisita che, in caso contrario, sarebbe immediatamente scartata. Nella situazione favorevole in cui venga effettivamente trovato uno *starting point* migliore, il controllo torna all'algoritmo ICP che effettua l'allineamento vero e proprio.

Il primo passo dell'algoritmo consiste nel generare un insieme di sei nuovi *starting point*, ognuno dei quali differisce dall'originale per una differente traslazione su un asse di riferimento, calcolando al tempo stesso la relativa funzione errore. Se già in questa prima fase si identifica uno *starting point* in grado di soddisfare i requisiti per l'allineamento, cioè se la funzione errore risulta inferiore ad una determinata soglia, si interrompe il Simulated Annealing e si avvia immediatamente l'ICP, che riceve in ingresso lo *starting point* da cui iniziare le operazioni di allineamento.

Altrimenti, si avvia la seconda fase dell'algoritmo: si seleziona lo *starting point* con errore associato maggiore e si effettua per esso una perturbazione alla relativa matrice di traslazione attraverso una distribuzione di probabilità gaussiana. Il valore della varianza della gaussiana, quindi dell'ampiezza della perturbazione, dipende dalla temperatura del sistema, che diminuisce o aumenta in base al valore della funzione errore e del numero di perturbazioni consecutive senza successo effettuate sullo *starting point* in esame. In seguito ad una perturbazione infatti, se uno *starting point* peggiora il suo stato, la temperatura aumenta per consentire un movimento più ampio. In base alla temperatura dipende perciò la probabilità di effettuare una perturbazione più o meno incisiva. Successivamente si ricalcola la funzione errore e, anche in questo caso, se il valore è inferiore ad una determinata soglia si riavvia l'ICP, altrimenti si controlla la variazione del valore della funzione errore: se esso è diminuito si reiterano le operazioni descritte e si procede con un'ulteriore perturbazione del nuovo *starting point* peggiore; altrimenti si ripristina lo *starting point* precedente.

Al fine di interrompere l'algoritmo nel caso in cui tutti gli *starting point* non sono in grado di trovare una buona convergenza, è stato introdotto un valore limite sul numero di perturbazioni effettuate per ogni *starting point*: se si oltrepassa tale valore, lo *starting point* in esame non è più considerato, altrimenti si procede con un'ulteriore perturbazione. Se nessuno *starting point* è considerato idoneo, si interrompe prematuramente l'algoritmo; altrimenti si reitera il procedimento. L'interruzione dell'algoritmo tuttavia non compromette l'intero processo di creazione del modello digitale; se il Simulated Annealing non riesce a fornire all'ICP uno starting point alternativo per allineare il fotogramma al modello, il fotogramma viene semplicemente rigettato, come avviene per la versione originale dell'ICP, ed il controllo torna all'ICP stesso, che riceverà in ingresso dallo scanner un nuovo fotogramma da allineare al modello.


(a)


(b)

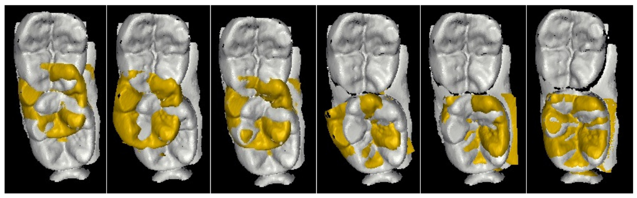

(c)

Fig. 3. (a) Sequenza rototraslazioni ICP (fallimento algoritmo originale)**;** (b) Nuvola di starting point (prima fase Simulated Annealing)**;** (c) Perturbazioni favorevoli del Simulated Annealing e successo dell'algoritmo.

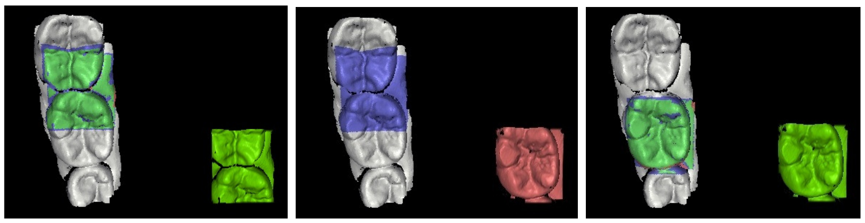


Fig. 4. (a) Ultimo allineamento corretto che precede la perdita di continuità; (b) ICP: allineamento del fotogramma non riuscito; (c) Simulated Annealing: allineamento del fotogramma riuscito.

3 Sperimentazione e risultati

Al fine di verificare i miglioramenti apportati dall'integrazione del Simulated Annealing e ICP per il caso considerato, sono stati raccolti ed esaminati diversi *dataset* di acquisizioni effettuate attraverso lo scanner intra-orale. I dati raccolti al fine di valutare l'adeguatezza dell'algoritmo si sono concentrati sull'analisi dei seguenti parametri: numero di situazioni in cui l'Iterative Closest Point non riesce a convergere adeguatamente; numero di successi del Simulated Annealing, sia nella prima fase di generazione della nuvola di *starting point*, sia nella seconda fase in cui vengono effettuate le perturbazioni; numero di fallimenti del Simulated Annealing; numero di fallimenti dell'ICP in seguito a successi del Simulated Annealing; tempistiche di esecuzione. La sperimentazione effettuata ha condotto ai seguenti risultati:

- nella maggior parte delle situazioni il numero di successi dell'algoritmo è nettamente superiore al numero di fallimenti;
- i maggiori benefici apportati sono pertinenti a perdite di continuità dovute a scostamenti del dispositivo che variano principalmente lo stato della matrice di traslazione, infatti il metodo non modifica gli angoli di rotazione;
- l'algoritmo implementato riesce a trovare già nella prima fase, in numerose situazioni, uno *starting point* adeguato da fornire all'ICP;
- in rare circostanze si verificano situazioni in cui nonostante il Simulated Annealing ha apparentemente trovato uno *starting point* adatto al corretto allineamento del fotogramma, l'ICP non è in grado di raggiungere la convergenza globale. Tuttavia ciò non influenza negativamente l'accuratezza del modello finale;
- in caso di successo nella prima fase, l'algoritmo converge molto rapidamente, in media circa 30 ms, mentre nella seconda fase il tempo medio risulta essere all'incirca di un ordine di grandezza superiore. Le situazioni peggiori si riscontrano in caso di fallimento dell'algoritmo, in cui il tempo si attesta attorno a 1500 ms (i valori devono essere considerati solamente per un'analisi della correttezza dell'algoritmo in quanto sono stati rilevati mediante un PC con processore Core Duo 2.20GHz, mentre i requisiti minimi richiederebbero un processore Core Duo 2.40GHz).

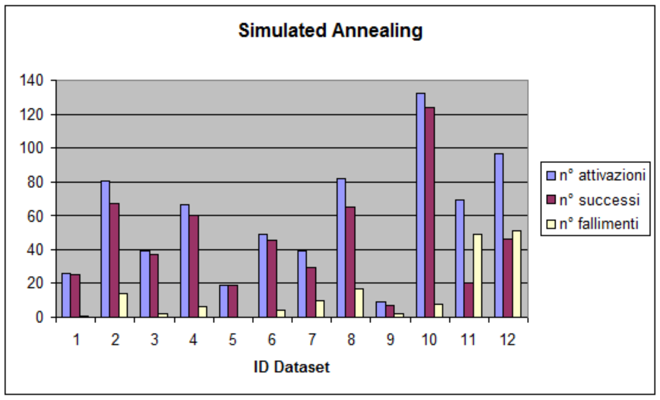


Fig. 5. Simulated Annealing: attivazioni, successi e fallimenti

4 Conclusione

L'integrazione della tecnica Simulated Annealing con l'algoritmo ICP si è rivelata un'ottima soluzione per far fronte al problema delle perdite di continuità dello scanner intra-orale considerato. In particolare, grazie all'identificazione del minimo globale di una funzione, Simulated Annealing è in grado di subentrare egregiamente nelle situazioni in cui l'Iterative Closest Point fallisce. Il metodo si è rivelato affidabile ed in grado di evitare le perdite di continuità nella maggioranza delle situazioni esaminate, condizione necessaria al fine di migliorare l'usabilità del dispositivo medico.

Tuttavia, al fine di rendere l'applicazione sufficientemente rapida per un utilizzo real-time, è consigliato l'utilizzo di tale algoritmo attraverso una procedura che opera in background al fine di evitare situazioni di interruzione temporanea generate dalla complessità di calcolo dell'algoritmo, evitando pertanto di sospendere la sequenza di acquisizioni. Il processo deve essere celato all'utilizzatore allo scopo di far recuperare la continuità automaticamente mentre lo scanner è in fase di acquisizione continua, senza quindi richiedere un intervento a livello fisico da parte dell'utilizzatore.

Riferimenti bibliografici

1. Kirkpatrick, S., Gelatt, C. D., Vecchi M. P., "Optimization by Simulated Annealing," Science, New Series, Vol. 220, No. 4598, 1983.
2. Chen, Y. and Medioni, G. "Object Modeling by Registration of Multiple Range Images," Proc. IEEE Conf. on Robotics and Automation, 1991.
3. Luck, J., Little, C., Hoff, W., "Registration of Range Data Using Hybrid Simulated Annealing and Iterative Closest Point," Proc. Of IEEE International Conference of Robotics and Automation, San Francisco, April 24-28, 2000.

I^2AM: a Semi-Automatic System for Data Interpretation in Petroleum Geology

Denis Ferraretti[1], Giacomo Gamberoni[1], and Evelina Lamma[2]

[1] intelliWARE snc, via Borgo dei Leoni 132, 44121 Ferrara, Italy
{denis, giacomo}@i-ware.it
[2] Engineering Department, University of Ferrara, via Saragat 1, 44122 Ferrara, Italy
{evelina.lamma}@unife.it

1 Introduction

The natural complexities of petroleum reservoir systems continue to provide a challenge to geoscientists. In petroleum geology, exploration and production wells are often analysed using image logs and the use of all the available borehole data to completely characterize the reservoir potentials and performance is an important task. The development of reliable interpretation methods is of prime importance regarding the reservoir understanding and data integration is a crucial step in order to create useful description models and to reduce the amount of time necessary for each study. Artificial intelligence, data mining techniques and statistical methods are widely used in reservoir modelling, for instance in prediction of sedimentary facies[3].

The aim of our work was to define and implement a suite of tools for interpretation of image logs and large datasets of subsurface data coming from geological exploration. This led to the development of **I^2AM** (Intelligent Image Analysis and Mapping), a semi-automatic system that exploits image processing algorithms and artificial intelligence techniques to analyse and classify borehole data. More in detail, the objectives of the **I^2AM** approach are: (1) to automatically extract rock properties information from all the different types of data recorded/measured in the wells, and visual features from image logs in particular; (2) to identify clusters along the wells that have similar characteristics; (3) to predict class distribution over new wells in the same area.

The main benefits of this approach are the ability to manage and use a large amount of subsurface data simultaneously. Moreover, the automatic identification of similar portions of wells by hierarchical clustering saves a lot of time for the geologist (since he analyses only the previously identified clusters). The interpretation time reduces from days to hours and subjectivity errors are avoided. Moreover, chosen clusters are the input for supervised learning methods which learn a classification that can be applied to new wells.

[3] A facies is a body of sedimentary rock distinguished from others by its lithology, geometry, sedimentary structures, proximity to other types of sedimentary rock, and fossil content.

2 The I^{2}AM Approach

With our system, we propose a cascade of techniques, i.e., pattern recognition, clustering and learning classifications algorithms, in order to:

1. first, automatically identify relevant features in image logs, by applying machine vision algorithms;
2. second, cluster several regions of the same well or of different wells into similar groups, by applying hierarchical clustering and choose the set of most significant clusters: this is done by the expert of the domain;
3. finally, feed a machine learning algorithm in order to learn a classifier to be applied to new instances and wells, possibly co-located.

See Figure 1 for the entire workflow.

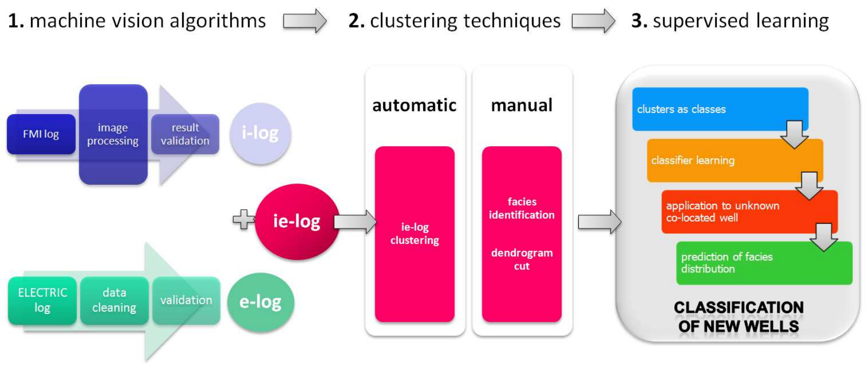


Fig. 1. Workflow of the I^{2}AM system: 1) image logs are analysed using machine vision algorithms and then merged with electrical logs; 2) clustering of the dataset in order to discover hidden data structures; 3) learning and classification of new wells.

In the first step we create a large dataset that includes data from different wells in the same area, this will be the input of following step. Each well is characterized by two types of log: image and electric. In order to use both we need to convert image log observations in numerical dataset. To do this we use machine vision algorithms.

In second step, hierarchical clustering is applied to a set of co-located wells in order to find an hidden data structure. The domain expert chooses the best clustering partition that fits the observed facies distribution. In our application we use hierarchical agglomerative clustering that produces a cluster hierarchy represented in a dendrogram. Using the dendrogram the geologist can choose the most suitable cluster partition.

Then in third step, starting from identified clusters, a supervised learning algorithm is used to learn a classifier which can be applied to new wells, in order to predict the distribution of facies over a new, unknown well in the same area. This task is achieved by learning the model of each cluster from the previous description, to this purpose it is possible to use different supervised techniques.

Following these steps, we obtain a semi-automatic interpretation and prediction method for well logs. This is a semi-automatic approach because a human quality control is needed in order to obtain a meaningful clustering partition in the domain context; but this is also the main advantage: the geologist identifies clusters only once considering all the available data simultaneously and saving time.

2.1 Machine Vision Algorithms

Image logs or FMI[4] logs are digital images acquired by a special logging tool within a borehole [14]. See Figure 2 for an example. FMI logs interpretation is a very complex task, due to the large number of variables and to the huge amount of data to be analysed. Usually, the geologist (domain expert) performs the bedding and fracture analysis by hand, in a tedious and expensive task, and then he tries to identify different classes that group well sections at different depths with similar visual characteristics.

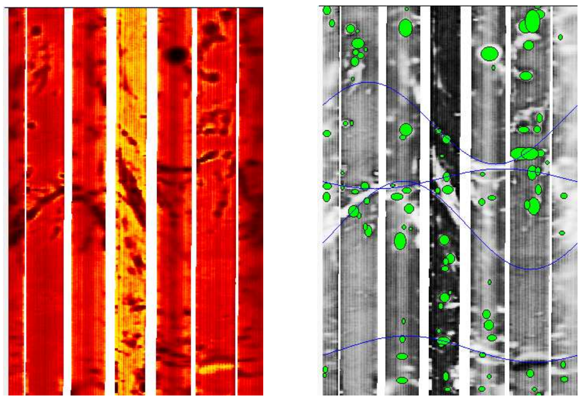

(a) Source FMI image (b) Automatic detection

Fig. 2. Example of FMI image log (a) and automatic extracted features (b): sinusoids (blue curves) and vacuoles (green circles).

The $\mathbf{I^2AM}$ approach for geological image interpretation is based on the detection/measurement of some features for each analysis window (360x100 pixel image), over the entire well.In particular these four features are:

[4] FMI (Fullbore Formation MicroImager) is the name of the tool used to acquire image logs based on resistivity measures within the borehole.

– surfaces (bedding or fracturing that visually correspond to sinusoids);
– vugs/clasts;
– contrast between the previous features and background;
– organization of the texture (homogeneous vs. granular).

In order to classify the features of the images over the entire well, the system analyzes the entire borehole log using an *analysis window* of fixed size. The size of the window is important because it has a direct impact on the resolution of the output/analysis and on the time of analysis of the entire well. The size of this window can be set by the user depending on the type of analysis to be performed.

Sinusoids in the log image can have different geological meanings: bedding or fracture. They do not appear entirely in the FMI, only short parts of them are directly visible.Sinusoids in the log image can have different geological meanings and they are automatically extracted using advanced image processing algorithms developed and tested in in [2] and [3].

To find and count vugs/clasts is important to understand the rock porosity and type of fluid that fills the vacuoles. In the FMI image vacuoles appear as circular or ellipsoidal areas with uniform color, with a high or low contrast with the background. To automatically find and count vugs/clasts the system use algorithms from [4]. The goal is to separate vacuoles from the background and to distinguish them on the basis of some visual features (i.e., area dimension or average color). A trivial count of the vacuoles and sinusoids detected in a zone are fundamental features for the classification of the rock.

The contrast value is significant because it can easily highlight the variation of resistivity in the rock formation. The resistivity variation usually depends on the lithology and the type of rock or type of fluids that fill the pores. This is achieved by using a properly filtered image FFT (Fast Fourier Transform), in order to link to each analyzing window a value that can represent a reliable measure of image contrast.

The internal organization of a rock is an important parameter to understand petrophysics and petrographic characteristics of a rock. The texture organization is highly variable and is an important information for the full interpretation of rock formation, it can be fine-grained to coarse-grained. A grainy FMI image has several small areas (grains) in contrast with the background, and these areas could be highlighted through an edge detection algorithm. The total amount of pixels in the edges of the processed image, is proportional to the texture organization.

Once the system has analysed the entire image log, and the algorithms have extracted the values that represent each feature, these information are summarized in a feature table (a row for each analysis window, a column for each image feature). This table is the final numerical dataset from FMI log and it can be properly merged with other electric logs.

2.2 Clustering Techniques

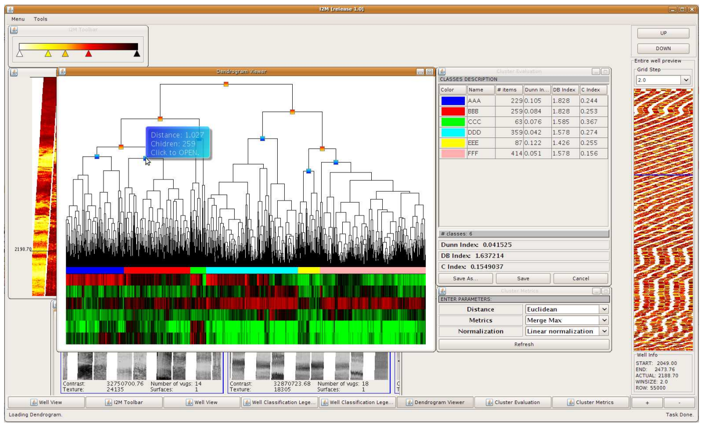
Cluster analysis is an unsupervised learning method that constitutes a cornerstone of our intelligent data analysis process [10]. It is defined as the task of categorizing objects having several attributes into different classes such that the objects belonging to the same class are similar, and those that are broken down into different classes are not. Intra-connectivity is a measure of the density of connections between the instances of a single cluster. A high intra-connectivity indicates a good clustering arrangement because the instances grouped within the same cluster are highly dependent on each other. Inter-connectivity is a measure of the connectivity between distinct clusters. A low degree of inter-connectivity is desirable because it indicates that individual clusters are largely independent of each other. Every instance in the dataset is represented using the same set of attributes.

Generally, clustering algorithms can be categorized into partitioning methods, hierarchical methods, density-based methods, and grid-based methods. In our work we use hierarchical method, it builds the hierarchy starting from the individual elements considered as single clusters, and progressively merges clusters according to a chosen similarity measure defined in features space. Hierarchical clustering techniques use various criteria to decide "locally" at each step which clusters should be joined (or split for divisive approaches). For agglomerative hierarchical techniques, the criterion is typically to merge the "closest" pair of clusters, where "close" is defined by a specified measure of cluster proximity. There are three definitions of the closeness between two clusters: single-link, complete-link and average-link. The single-link similarity between two clusters is the similarity between the two most similar instances, one of which appears in each cluster. Single link is good at handling non-elliptical shapes, but is sensitive to noise and outliers. The complete-link similarity is the similarity between the two most dissimilar instances, one from each cluster. Complete link is less susceptible to noise and outliers, but can break large clusters, and has trouble with convex shapes. The average-link similarity is a compromise between the two. Our application provides best known distance measures: Pearson, Manhattan and Euclidean, and linkage strategies (single, complete and average).

Results of agglomerative algorithms can be represented by dendrograms (see main windows in Figure 3). Advantages of this technique are: 1) it does not require the number of clusters to be known in advance, 2) it computes a complete hierarchy of clusters, 3) good result visualizations are integrated into the methods, 4) a "flat" partition can be derived afterwards (e.g. via a cut through the dendrogram). An excellent survey of clustering techniques can be found in [8].

2.3 Supervised Learning

Inductive machine learning is the process of learning a set of rules from instances (examples in a training set), or more generally speaking, creating a classifier that can be used to generalize from new instances [9].


CLASSES DESCRIPTION

Color	Name	# items	Dunn In...	DB Index	C Index
🟦	AAA	229	0.105	1.828	0.244
🟥	BBB	259	0.084	1.828	0.253
🟩	CCC	63	0.076	1.585	0.367
🟦	DDD	359	0.042	1.578	0.274
🟨	EEE	87	0.122	1.426	0.255
🟥	FFF	414	0.051	1.578	0.156

classes: 6

Dunn Index: 0.041525

DB Index: 1.637214

C Index: 0.1549037

ENTER PARAMETERS:

Distance	Euclidean
Metrics	Merge Max
Normalization	Linear normalization

Well Info
START: 2049.00
END: 2473.76
ACTUAL: 2188.70
WINSIZE: 2.0
ROW: 55000

Distance: 1.027
Children: 259
Click to OPEN

Fig. 3. Clustering process with dendrogram visualization in the I^{2}AM software.

Supervised classification is one of the tasks most frequently carried out by so-called Intelligent Systems. Thus, a large number of techniques have been developed based on artificial intelligence (logical/symbolic techniques), perceptron based techniques and statistics (bayesian networks, instance-based techniques).

In order to find the best classifier for facies distribution prediction in petroleum geology domain, we test several algorithms: decision trees, classification rules and regression methods. These techniques allow the propagation of classes to new wells. We use `J4.8`, `Random Forests`, `PART` [5] and `Rotation Forest` as decision trees induction and classification rules generation algorithms. For regression we use `ClassificationViaRegression` [6] and `Logistic`.

Decision trees represent classification rules in form of a tree, where each node represents a test on an attribute. Depending on the outcome of the test, we must follow the relative branch, and continue until we reach a leaf, that gives a classification of the instance. Decision trees are usually created from examples, using algorithms such as `C4.5` by Quinlan [12]. We use `J4.8` algorithm, which is an implementation of this `C4.5` decision tree learner.

`Random Forests` are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest [1]. The generalization error for forests converges to a limit as the number of trees in the forest becomes large.

`Rotation Forest` is an algorithm for generating ensembles of classifiers [13]. It consists in splitting the feature set into K subsets, running principal component analysis separately on each subset and then reassembling a new extracted feature set while keeping all the components. The data is transformed linearly into the new features. A decision tree classifier is trained with this data set.

Linear regression can easily be used for classification in domains with numeric attributes. Indeed, we can use any regression technique, whether linear or nonlinear, for classification. The trick is to perform a regression for each class, setting the output equal to one for training instances that belong to the class and zero for those that do not. The result is a linear expression for the class. Then, given a test example of unknown class, calculate the value of each linear expression and choose the one that is largest. This method is sometimes called *multiresponse linear regression*. We use `Logistic`, an implementation of a two-class logistic regression model with a ridge estimator [11]. A complete review of supervised machine learning techniques can be found in [9].

All supervised learning techniques were tested using WEKA, the open source data mining software written in Java [7]. Using several evaluation techniques, detailed in [3], we test classes prediction for 2 wells in a group of 6, and `Logistic` shows better performance than other algorithms in most cases. This result confirm, as expected, that regression methods are suitable for prediction of continuous numeric values.

References

1. L. Breiman, Random Forests, *Machine Learning*, 2001, pp. 5–32.
2. D. Ferraretti, *Data Mining for Petroleum Geology*, PhD Thesis, University of Ferrara, Italy, 2012.
3. D. Ferraretti and G. Gamberoni and E. Lamma, *Unsupervised and supervised learning in cascade for petroleum geology*, Expert Systems with Applications, Elsevier, 2012.
4. D. Ferraretti, Casarotti L., Gamberoni G., E. Lamma, Spot detection in images with noisy background, 16th International Conference on Image Analysis and Processing (ICIAP), Ravenna, Italy, 2011.
5. E. Frank and I. H. Witten, Generating Accurate Rule Sets Without Global Optimization, Proceedings of the Fifteenth International Conference on Machine Learning, 1998, pp. 144–151.
6. E. Frank, Y. Wang, S. Inglis, G. Holmes and I. H. Witten, Using Model Trees for Classification, *Machine Learning*, 1998, pp. 63–76.
7. M. Hall and E. Frank and G. Holmes and B. Pfahringer and P. Reutemann and I.H. Witten, *The WEKA data mining software: an update*, ACM SIGKDD Explorations Newsletter, Vol. 11, No. 1, pp. 10–18, ACM, 2009.
8. A.K. Jain and M.N. Murty and P.J. Flynn, *Data clustering: a review*, ACM computing surveys (CSUR), Vol. 31, No. 3, pp. 264–323, ACM, 1999.
9. S.B. Kotsiantis and I.D. Zaharakis and P.E. Pintelas, *Supervised machine learning: A review of classification techniques*, Frontiers in Artificial Intelligence and Applications, Vol. 160, IOS Press, 2007.
10. S. B. Kotsiantis and P.E. Pintelas, *Recent advances in clustering: A brief survey*, WSEAS Transactions on Information Science and Applications, Vol. 1, No. 1, pp. 73–81, Citeseer, 2004.
11. S. Le Cessie, J. C. Van Houwelingen, Ridge Estimators in Logistic Regression, *Applied Statistics*, 1992.
12. J. R. Quinlan, Induction on Decision Trees, *Machine Learning*, 1986, pp. 81–106.
13. J. J. Rodriguez, L. I. Kuncheva, C. J. Alonso, Rotation Forest: A New Classifier Ensemble Method, *IEEE Trans. Pattern Anal. Mach. Intell.*, 2006, pp. 1619–1630.
14. O. Serra and L. Serra, *Well Logging, Data Acquisition and Applications*, SerraLog, 2004.

TRENTINOMEDIA: Exploiting NLP and Background Knowledge to Browse a Large Multimedia News Store[⋆]

Roldano Cattoni[1], Francesco Corcoglioniti[1,2], Christian Girardi[1], Bernardo Magnini[1], Luciano Serafini[1], and Roberto Zanoli[1]

[1] Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy
[2] DISI, University of Trento, Via Sommarive 14, 38123 Trento, Italy

{cattoni,corcoglio,cgirardi,magnini,serafini,zanoli}@fbk.eu

Abstract. TRENTINOMEDIA provides access to a large and daily updated repository of multimedia news in the Trentino Province. Natural Language Processing (NLP) techniques are used to automatically extract knowledge from news, which is then integrated with background knowledge from (Semantic) Web resources and exploited to enable two interaction mechanisms that ease information access: entity-based search and contextualized semantic enrichment. TRENTINOMEDIA is a real multimodal archive of public knowledge in Trentino and shows the potential of linking knowledge and multimedia and applying NLP on a large scale.

1 Introduction

Finding information about an entity in a large news collection using standard keyword-based search may be time-consuming. Searching for a specific person, for instance, may return a large list of news about homonymous persons, that need to be checked and filtered manually. Also, understanding the contents of a news can be expensive, if the user is not familiar with the entities mentioned and needs information about them.

The presented TRENTINOMEDIA system shows how the use of NLP and Semantic Web techniques may help in addressing these problems. TRENTINOMEDIA supports the "smart" access to a large and dynamic (daily updated) repository of multimedia news in the Italian Trentino Province. "Smart" means that NLP techniques are used to automatically extract knowledge about the entities mentioned in the news. Extracted knowledge is then integrated with *background knowledge* about the same entities gathered from (Semantic) Web resources, so to build a comprehensive knowledge base of entity descriptions linked to the news of the collection. Exploiting the interlinking of knowledge and multimedia, two interaction mechanisms are provided to ease information access:

- *entity-based search*, enabling a user to find exactly the news about a specific entity;
- *contextualized semantic enrichment*, consisting in the visualization of additional knowledge about a mentioned entity that may ease a user's understanding of a news.

Two main usages are foreseen for TRENTINOMEDIA: (i) a professional usage, restricted to the news providers and aimed at addressing internal needs, including automatic news documentation, support tools for journalists and integration of advanced

[⋆] This work was supported by the LiveMemories project (http://www.livememories.org) funded by the Autonomous Province of Trento (Italy) under the call "Major Project".

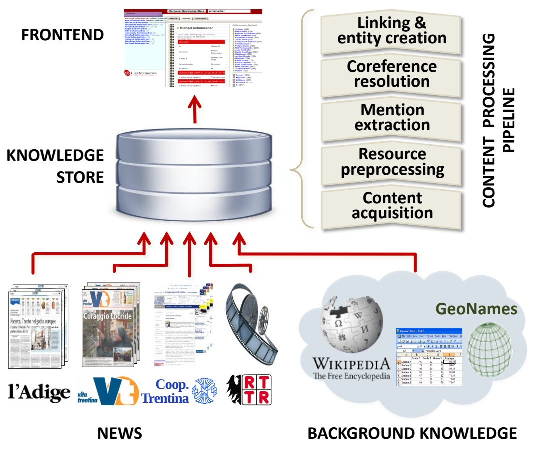


Fig. 1: System architecture

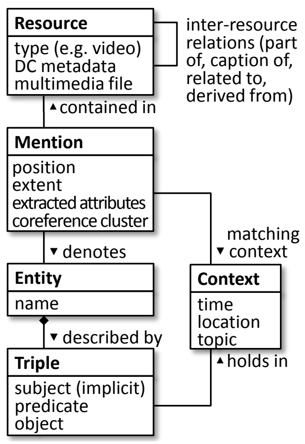


Fig. 2: Data model

functionalities in existing editorial platforms; and (ii) an open use by citizens through on-line subscriptions to news services, possibly delivered to mobile devices.

The presented work has been carried out within the LiveMemories project, aimed at automatically interpreting heterogeneous multimedia resources, transforming them into "active memories". The remainder of the paper presents the system architecture in section 2 and the demonstrated user interface in section 3, while section 4 concludes.

2 System architecture

The architecture of TRENTINOMEDIA is shown in figure 1 and includes three components: the KNOWLEDGESTORE, a *content processing pipeline* and a *Web frontend*.

The KNOWLEDGESTORE [3] builds on Hadoop[3] and Hbase[4] to provide a scalable storage for multimedia news and background knowledge, which are represented according to the (simplified) schema of figure 2. News are stored as multimedia *resources*, which include texts, images and videos. Knowledge is stored as a *contextualized ontology*. It consists of a set of *entities* (e.g., "Michael Schumacher") which are described by ⟨subject, predicate, object⟩ RDF [2] *triples* (e.g., ⟨Michael Schumacher, pilot of, Mercedes GP⟩). In turn, each triple is associated to the ⟨time, space, topic⟩ *context* the represented fact holds in (e.g., ⟨2012, World, Formula 1⟩). The representation of contexts follows the Contextualized Knowledge Repository approach [5] and permits to accommodate "conflicting" knowledge holding under different circumstances (e.g. the fact that Schumacher raced for different teams). Resources and entities are linked by *mentions*, i.e. proper names in a news that refer to an entity. They permit to navigate from a news to its mentioned entities and back, realizing the tight interlinking of knowledge and multimedia at the basis of the interaction mechanisms of TRENTINOMEDIA.

Concerning the content processing pipeline, it integrates a number of NLP tools to load, process and interlink news and background knowledge, resulting in the full

[3] http://hadoop.apache.org
[4] http://hbase.apache.org

Table 1: Resource statistics

Provider	News	Images	Videos
l'Adige	733,738	21,525	-
VitaTrentina	33,403	14,198	-
RTTR	2,455	-	120 h
Fed. Coop.	1,402	-	-
Total	770,998	35,723	120 h

Table 2: Extraction, coref. and linking stats.

Entity type	Recognized mentions	Mention clusters	Linked clusters	Total entities
persons	5,566,174	340,147	5.03%	351,713
organiz.	3,230,007	16,649	7.96%	17,129
locations	3,224,539	52,478	48.64%	52,478
Total	12,020,720	409,274	10.74%	421,320

population of the schema in figure 2. Apart from the loading of background knowledge, which is bootstrapped by manually selecting and wrapping the relevant knowledge sources, the pipeline works automatically and incrementally, processing news as they are collected daily. The rest of this section describes the processing steps of the pipeline, while the user interface of the frontend is described in the next section.

Content acquisition. News are supplied daily by a number of news providers local to the Trentino region. They are in Italian, cover a time period from 1999 to 2011 and consist of text articles, images and videos. Loading of news is performed automatically and table 1 shows some statistics about the news collected so far. Background knowledge is collected manually through a set of ad-hoc wrappers from selected (Semantic) Web sources, including selected pages of the Italian Wikipedia, sport-related community sites and the sites of local and national public administrations and government bodies. Overall, it consists of 352,244 facts about 28,687 persons and 1,806 organizations.

Resource preprocessing. Several operations are performed on stored news with the goal of easing their further processing in the pipeline. Preprocessing includes the extraction of speech transcription from audio and video news, the annotation of news with a number of linguistic taggers (e.g., part of speech tagging and temporal expression recognition, performed using the TextPro tool suite[5] [6]) and the segmentation of complex news in their components (e.g., the separation of individual stories in a news broadcast or the extraction of texts, figures and captions from a complex XML article).

Mention extraction. Textual news are processed with TextPro to recognize mentions of three types of named entities: persons, organizations and geo-political / location entities. For each mention, a number of attributes is extracted from the mention and its surrounding text. Given the text "the German pilot Michael Schumacher", for instance, the system recognizes "Michael Schumacher" as a person mention and annotates it with FIRSTNAME "Michael", LASTNAME "Schumacher", ROLE "pilot" and NATIONALITY "German". Attributes are extracted based on a set of rules (e.g., to split first and last names) and language-specific lists of words (e.g., for nationalities, roles, . . .). Statistics about the mentions recognized so far are reported in the second column of table 2.

Coreference resolution. This step consists in grouping together in a *mention cluster* all the mentions that (are assumed to) refer to the same entity, e.g., to decide that mentions "Michael Schumacher" and "Schumi" in different news denote the same person. Two coreference resolution systems are used. Person and organization mentions are processed with JQT2 [8], a system based on the Quality Threshold algorithm [4] that

[5] http://textpro.fbk.eu

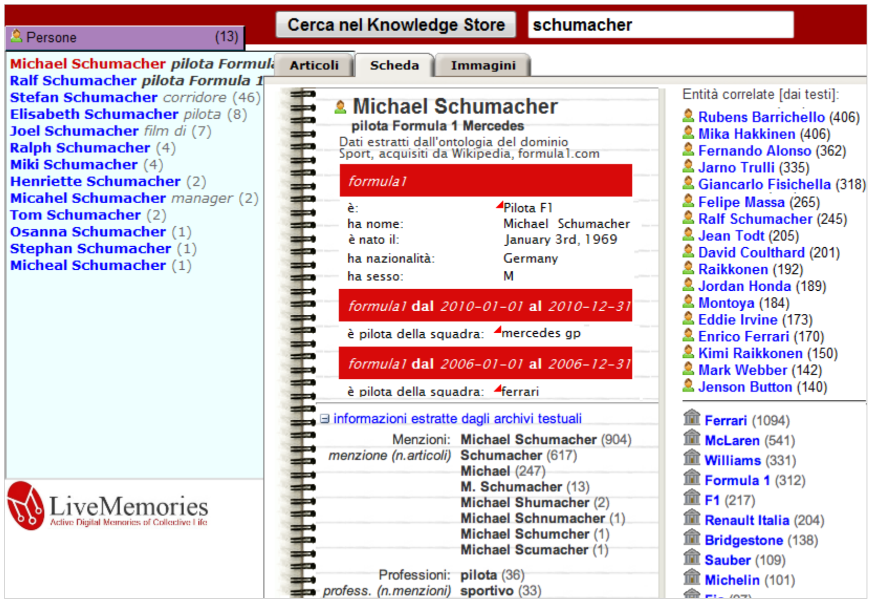
compares every pair of mentions and decides for coreference if their similarity score is above a certain *dynamic threshold*; similarity is computed based on a rich set of features (e.g., mention attributes and nearby words), while the threshold is higher for ambiguous names, requiring more "evidence" to assume coreference. Location mentions are processed with GeoCoder [1], a system based on geometric methods and on the idea that locations in the same news are likely to be close one to another; it exploits the Google Maps geo-referencing service[6] and the GeoNames geographical database[7]. Statistics about the mention clusters identified so far are reported in the third column of table 2.

Linking and entity creation. The last step consists in linking mention clusters to entities in the background knowledge and to external knowledge sources. Clusters of location mentions are already linked to well-known GeoNames toponyms by GeoCoder. Clusters of person and organization mentions are linked to entities in the background knowledge by exploiting the representation of contexts. The algorithm [7] firstly identifies the ⟨time, space, topic⟩ contexts most appropriate for a mention cluster among the ones in the KNOWLEDGESTORE, based on the mentions attributes and the metadata of the containing news (e.g., the publication date). Then, it searches for a matching entity only in those contexts, improving disambiguation. The fourth column of table 2 reports the fraction of mention clusters linked by the systems, i.e. the *linking coverage*: coverage is low for clusters (10.74%), but increases in terms of mentions (31.03%), meaning that the background knowledge mainly consists of popular (and thus frequently mentioned) entities. New entities are then created and stored for unlinked mention clusters, as they denote real-world entities unknown in the background knowledge; the last column of table 2 reports the total number of entities obtained so far. All the entities are finally associated to the corresponding Wikipedia pages using the WikiMachine tool[8].

3 User Interface

The entry point of the TRENTINOMEDIA Web interface is a search page supporting *entity-based search*. The user supplies a proper name which is looked up among the entities in the KNOWLEDGESTORE and a list of matching entities is returned for disambiguation; entities are listed by type and distinguished with short labels generated from stored information, as in figure 3, left side. By selecting an entity, the user is presented with the list of news mentioning that entity, retrieved based on the associations between entities, mentions and resources stored in the KNOWLEDGESTORE. A descriptive card is also displayed, as shown in the right side of figure 3. It contains all the information known about the entity, including: (i) background knowledge, (ii) information carried by the attributes of the entity mentions and (iii) frequently co-occurring and likely related entities. The example in figure 3 shows the potential but also the weaknesses of processing noisy, real world data with automatic NLP tools. In particular, typos and the use of different names for the same entity (e.g., acronyms, abbreviations) may cause coreference resolution to fail and identify multiple entities in place of one, as happens with "F1" and "Formula 1", "Raikkonen" and "Kimi Raikkonen", "Mich*ea*l Schumacher" and "Mich*ae*l Schumacher". Still, the use of additional information ex-

[6] http://code.google.com/apis/maps/
[7] http://www.geonames.org/
[8] http://thewikimachine.fbk.eu/



Cerca nel Knowledge Store | schumacher

Persone (13)

Michael Schumacher *pilota Formul*
Ralf Schumacher *pilota Formula 1*
Stefan Schumacher *corridore* (46)
Elisabeth Schumacher *pilota* (8)
Joel Schumacher *film di* (7)
Ralph Schumacher (4)
Miki Schumacher (4)
Henriette Schumacher (2)
Micahel Schumacher *manager* (2)
Tom Schumacher (2)
Osanna Schumacher (1)
Stephan Schumacher (1)
Micheal Schumacher (1)

LiveMemories
Active Digital Memories of Collective Life

Articoli | Scheda | Immagini

Michael Schumacher
pilota Formula 1 Mercedes
Dati estratti dall'ontologia del dominio
Sport, acquisiti da Wikipedia, formula1.com

formula1

è:	Pilota F1
ha nome:	Michael Schumacher
è nato il:	January 3rd, 1969
ha nazionalità:	Germany
ha sesso:	M

formula1 **dal** *2010-01-01* **al** *2010-12-31*

è pilota della squadra: mercedes gp

formula1 **dal** *2006-01-01* **al** *2006-12-31*

è pilota della squadra: ferrari

⊟ informazioni estratte dagli archivi testuali

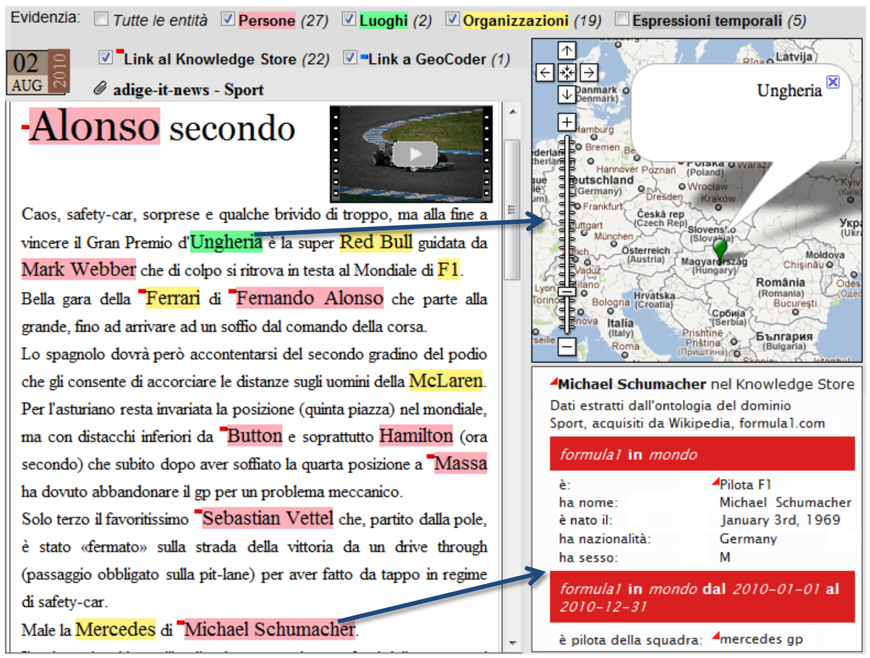
Menzioni:	**Michael Schumacher** (904)
menzione (n.articoli)	**Schumacher** (617)
	Michael (247)
	M. Schumacher (13)
	Michael Shumacher (2)
	Michael Schnumacher (1)
	Michael Schumcher (1)
	Michael Scumacher (1)
Professioni:	**pilota** (36)
profess. (n.menzioni)	**sportivo** (33)

Entità correlate [dai testi]:

Rubens Barrichello (406)
Mika Hakkinen (406)
Fernando Alonso (362)
Jarno Trulli (335)
Giancarlo Fisichella (318)
Felipe Massa (265)
Ralf Schumacher (245)
Jean Todt (205)
David Coulthard (201)
Raikkonen (192)
Jordan Honda (189)
Montoya (184)
Eddie Irvine (173)
Enrico Ferrari (170)
Kimi Raikkonen (150)
Mark Webber (142)
Jenson Button (140)

Ferrari (1094)
McLaren (541)
Williams (331)
Formula 1 (312)
F1 (217)
Renault Italia (204)
Bridgestone (138)
Sauber (109)
Michelin (101)

Fig. 3: Example of entity-based search for query “Schumacher” in TRENTINOMEDIA.



Evidenzia: ☐ *Tutte le entità* ☑ **Persone** *(27)* ☑ **Luoghi** *(2)* ☑ **Organizzazioni** *(19)* ☐ **Espressioni temporali** *(5)*

02 AUG 2010 ☑ **Link al Knowledge Store** *(22)* ☑ **Link a GeoCoder** *(1)*

adige-it-news - Sport

Alonso secondo

Caos, safety-car, sorprese e qualche brivido di troppo, ma alla fine a vincere il Gran Premio d'Ungheria è la super Red Bull guidata da Mark Webber che di colpo si ritrova in testa al Mondiale di F1.
Bella gara della Ferrari di Fernando Alonso che parte alla grande, fino ad arrivare ad un soffio dal comando della corsa.
Lo spagnolo dovrà però accontentarsi del secondo gradino del podio che gli consente di accorciare le distanze sugli uomini della McLaren.
Per l'asturiano resta invariata la posizione (quinta piazza) nel mondiale, ma con distacchi inferiori da Button e soprattutto Hamilton (ora secondo) che subito dopo aver soffiato la quarta posizione a Massa ha dovuto abbandonare il gp per un problema meccanico.
Solo terzo il favoritissimo Sebastian Vettel che, partito dalla pole, è stato «fermato» sulla strada della vittoria da un drive through (passaggio obbligato sulla pit-lane) per aver fatto da tappo in regime di safety-car.
Male la Mercedes di Michael Schumacher.

Michael Schumacher nel Knowledge Store
Dati estratti dall'ontologia del dominio
Sport, acquisiti da Wikipedia, formula1.com

formula1 **in** *mondo*

è:	Pilota F1
ha nome:	Michael Schumacher
è nato il:	January 3rd, 1969
ha nazionalità:	Germany
ha sesso:	M

formula1 **in** *mondo* **dal** *2010-01-01* **al** *2010-12-31*

è pilota della squadra: mercedes gp

Fig. 4: SmartReader showing a sport news.

tracted from texts (e.g., keywords) can often overcome the problem, as happens with the correct coreference of "Schumacher", "Michael" and "Michael S*hu*macher").

The other interaction mechanism supported by TRENTINOMEDIA—*contextual semantic enrichment*—is accessed through the *SmartReader* interface shown in figure 4, which is displayed by selecting a news. The SmartReader allows a user to read news or watch videos while gaining access to related information linked in the KNOWLEDGESTORE to the news and its mentioned entities. The interface is organized in two panels. The left panel displays the text of the news or the video with its speech transcription and permits to selectively highlight the recognized mentions of named entities. The right panel provides *contextual information* that enriches the news or a selected mention. It can display a cloud of automatically extracted keywords, each providing access to related news. It can also show additional information about the selected mention, by presenting: (i) the Wikipedia page associated to the mentioned entity, (ii) a map displaying a location entity and (iii) a descriptive card with information about the entity in the background knowledge. In the latter case, only facts which are valid in the $\langle$time, space, topic$\rangle$ context of the news are shown, e.g., that "Schumacher is a pilot of Mercedes GP in 2010", so to avoid to overload and confound the user with irrelevant information.

4 Conclusions

TRENTINOMEDIA shows how the application of NLP techniques and the interlinking of knowledge and multimedia resources can be beneficial to users accessing information contents. In particular, two mechanisms to exploit this interlinking are demonstrated: entity-based search exploits links from knowledge (entities) to resources, while semantic enrichment exploits links in the opposite direction. TRENTINOMEDIA also shows that NLP and Semantic Web technologies are mature enough to support the large scale extraction, storage and processing of knowledge from multimedia resources.

References

1. Buscaldi, D., Magnini, B.: Grounding toponyms in an Italian local news corpus. In: Proc. of 6th Workshop on Geographic Information Retrieval. pp. 15:1–15:5. GIR '10 (2010)
2. Carroll, J.J., Klyne, G.: Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation (2004), http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/
3. Cattoni, R., Corcoglioniti, F., Girardi, C., Magnini, B., Serafini, L., Zanoli, R.: The KNOWLEDGESTORE: an entity-based storage system. In: Proc. of 8th Int. Conf. on Language Resources and Evaluation. LREC '12 (2012)
4. Heyer, L.J., Kruglyak, S., Yooseph, S.: Exploring expression data: Identification and analysis of coexpressed genes. Genome Research 9(11), 1106–1115 (1999)
5. Homola, M., Tamilin, A., Serafini, L.: Modeling contextualized knowledge. In: Proc. of 2nd Int. Workshop on Context, Information And Ontologies. CIAO '10, vol. 626 (2010)
6. Pianta, E., Girardi, C., Zanoli, R.: The TextPro tool suite. In: Proc. of 6th Int. Conf. on Language Resources and Evaluation. LREC '08 (2008)
7. Tamilin, A., Magnini, B., Serafini, L.: Leveraging entity linking by contextualized background knowledge: A case study for news domain in Italian. In: Proc. of 6th Workshop on Semantic Web Applications and Perspectives. SWAP '10 (2010)
8. Zanoli, R., Corcoglioniti, F., Girardi, C.: Exploiting background knowledge for clustering person names. In: Proc. of Evalita 2011 – Evaluation of NLP and Speech Tools for Italian (2012), to appear

Multimodal interaction with emotional feedback

Francesco Cutugno[1], Antonio Origlia[1], and Roberto Rinaldi[1]

LUSI-lab, Department of Physics, University of Naples “Federico II”, Naples, Italy
cutugno@unina.it antonio.origlia@unina.it rober.rinaldi@studenti.unina.it

Abstract. In this paper we extend a multimodal framework based on speech and gestures to include emotional information by means of anger detection. In recent years multimodal interaction has become of great interest thanks to the increasing availability of mobile devices allowing a number of different interaction modalities. Taking *intelligent* decisions is a complex task for automated systems as multimodality requires procedures to integrate different events to be interpreted as a single intention of the user and it must take into account that different kinds of information could come from a single channel as in the case of speech, which conveys a user’s intentions using syntax and prosody both.

1 Introduction

Multimodal interaction involving speech aims at providing a more natural interactive experience to the users of automated systems. While this is indeed an important and ambitious goal, it introduces a number of error sources caused by the potential ambiguities that can be found in natural language and the performance of Automatic Speech Recognition (ASR). This has introduced the need for an automated system to perform some kind of self-monitoring to evaluate its own performance, detect erroneous task selection and avoid committing the same error two times in the same interactive session. This field has been deeply explored by researchers dealing with Interactive Voice Response (IVR) platforms, where speech is the only source of information the systems can use in order to select which one of the services it offers the user is looking to obtain and in order to collect the information needed to complete the requested task. While semantic content extraction is obviously the main cue to perform task selection and data collection, paralinguistic information has been used in IVR systems to perform self-evaluation and support interaction with the user [1,15]. These early systems were trained on acted emotions while recent research is now concentrating on spontaneoud emotions: in [4] a real life anger detector trained on data collected from a voice portal was presented while in [14] the problem of multilingual anger detection was explored using recordings from an IVR system.

2 State of the art

Multimodal interface systems were introduced for the first time in the system presented in [3], where graphical objects were created and moved on a screen using voice recognition and finger pointing. In [5] a set of theoretical guidelines were defined that were named CARE Proprieties (Complementary, Assignment, Redundancy, Equivalence). These properties establish which modes of interaction between users and systems can be implemented and, at the same time, help to formalize relationships among different modalities. The increasing amount of research and practical applications of multimodal interaction systems recently led to the definition of the Synchronized Multimodal User Interaction Modeling Language (SMUIML) [7]: a formal way of representing multimodal interactions. While the possibilities of implementing multimodal information access systems has been explored since when mobile phones started to offer internet based services [16], with the widespread adoption of touch screens on mobile devices, mobile broad band and fast speech recognition, interfaces supporting truly multimodal commands are now available to everyday users. An example is the *Speak4it* local search application [8], where users can use multimodal commands combining speech and gestures to issue mobile search queries. The great interest risen from the possibilities offered by this kind of systems, not only in a mobile environment, soon highlighted the need of formalizing the requirements an automated interactive systems needs to fulfill to be considered *multimodal*. This problem was addressed by the W3C, which has established a set of requirements, concerning both interaction design [11] and system architecture [2], formalized as proprieties and theoretical standards multimodal architectures

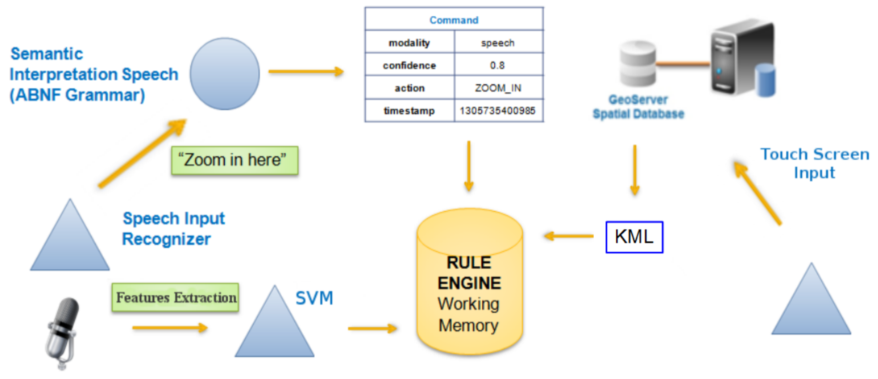
Concerning the use of anger detectors in IVRs, in previous studies [13,15] systems have been usually trained on acted emotions corpora before being deployed on IVR platforms. An exception to this trend is represented by [10], in which a corpus of telephone calls collected from a troubleshooting call-center database was used. In that study, the impact of emotions was shown to be minimal with respect to the use of log-files as the authors observed a uniform distribution of negative emotions over successful and unsuccessful calls. This, however, may be a characteristic of the employed corpus, in which people having problems with a High Speed Internet Provider were calling, and is therefore significantly different from the situation our system deals with, as our target consists of users of a bus stops information service.

3 System architecture

In this paper we extend a pre-existing multimodal framework, running on Android OS, based on speech and gesture to include emotional information by means of a user emotional attitude detector. We merge these concepts in a case study previously presented in [6], in which a querying system for bus stops in the city of Naples was implemented. Users can query the system by speaking and drawing on the touch screen producing requests for bus stops in a given area on the map. In a typical use case the user asks: *"Please show me the bus stops of C6 line in this area"* drawing a circle on a map on the screen while speaking.

The user can draw lines and circles on a map aiming at selecting a precise geographic area of interest concerning public transportation. In addition the user can hold her finger for some second on a precise point on the map in order to select a small rectangular default area on the map with the same purposes. At the same time, speech integrates the touch gesture to complete the command. This way, users can ask for a particular bus line or timetable (using speech) in a given geographic area (using touch), as shown in Figure 1.

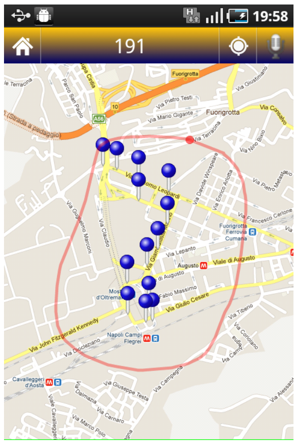
For details concerning the general architecture, we refer the reader to [6]. In the present system, the audio signal is considered as twin input: the first one connected to the linguistic content itself obtained by means of an ASR process and a subsequent string parsing process that generates a *Command* table structurally incomplete as more data are needed in correspondence with the missing geographical data completing the user request; the latter goes to an emotional attitude classifier (details will be presented in the next section) returning the anger level characterizing the utterance produced by the user.



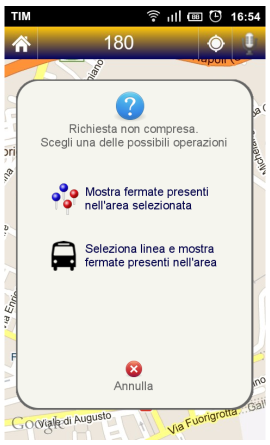
Command	
modality	speech
confidence	0.8
action	ZOOM_IN
timestamp	1305735400985

Fig. 1. System architecture: interaction example

The semantic interpreter collects the inputs from parsed ASR and from touch/geographical modules and attempts an answer using the freely available Drools (http://www.jboss.org/drools) rule engine while anger detection is used to launch backup strategies if the transaction does not succeed and the user is unsatisfied by the service as shown in Figure 2.



(a) Places of interest found by combining speech and gestures



Richiesta non compresa.
Scegli una delle possibili operazioni

Mostra fermate presenti nell'area selezionata

Seleziona linea e mostra fermate presenti nell'area

Annulla

(b) Backup strategy for unrecognized commands with angry users

Fig. 2. Screenshots of a multimodal interactive system on a mobile device. In 2a an example of a combined speech and gesture based interaction. Given the utterance "Tell me about the stops of the 191 line # *id_speech_xyz - here - #*" and the co-occurring gesture command # *id_gesture_xyz - drawCircle #*, points corresponding to bus stops of the 191 line are drawn in the area of interest. In 2b a popup menu is used as backup strategy when the transaction fails and the user is getting angry.

4 Emotion recognition module

Automatic emotion recognition is a research topic that has been gaining attention in the last years because of the additional information it brings into automatic systems about the users' state of mind. While there are a number of applications and representations of emotions in the literature, one that has found application in IVR systems is anger detection. Capturing a negative state of the speaker during the interaction is an information that has been exploited in the past, for example, in automated call centers to forward the call to a human agent. Anger detection is usually based on the response given by an automatic classifier on the basis of acoustic features extracted from a received utterance. Features extraction and classification methods for emotions are active research areas: in this work, we use a syllable-based features extraction method and a Support Vector Machine (SVM) to perform the automatic classification of an utterance into two classes: Neutral and Angry. The anger detection module is trained on a subpart of the €motion corpus [9] containing 400 angry and neutral speech recordings in Italian, German, French and English.

First, the recorded utterance is segmented into syllables. This is done by applying the automatic segmentation algorithm presented in [12]. Next, data are extracted from syllable nuclei, estimated by the -3db band of the energy peak associated with each automatically detected syllable. Syllable nuclei, being stable spectral areas containing vowel sounds, contain more reliable information regarding the distribution of the energy among the frequencies as it was intended by the speaker. Specific spectral measurements like the spectral centroid, moreover, do make sense inside syllable nuclei only. To improve the reliability of the extracted measures, only syllable nuclei at least 80ms long were analyzed. An example of automatic syllable nuclei detection is shown in Figure 3.

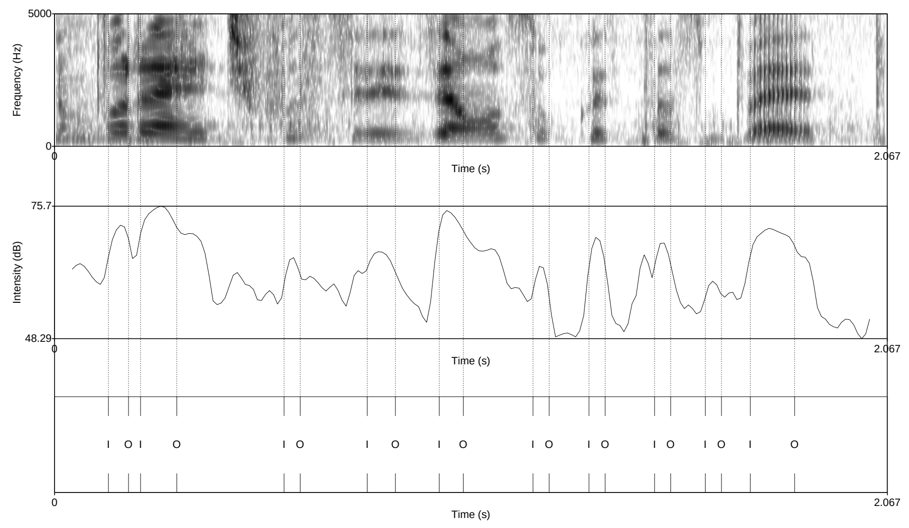


Fig. 3. Automatic detection of syllable nuclei. On the first level, the spectrogram of a speech utterance is shown. On the second one, the energy profile is reported while on the third one automatically detected syllable nuclei incipits (I) and offsets (O) are shown. Voiced areas of spectral stability are used to extract cleaner features.

From each nucleus we extract the following features: *mean pitch* (perceived fundamental frequency), *spectral centroid* (mean of the frequencies in the spectrum weighted by their magnitude) and *energy.*

To produce the final features set, global statistics were computed over the feature vectors extracted from each syllable. Mean and standard deviation were included for each feature while the maximum value was introduced for energy only. An SVM was trained and tested on the features extracted from the €motion corpus. The F-measure obtained in a 10-fold cross validation test was 90.5%.

5 Discussion

The proposed system is presently still under development so its usability has not yet been completely assessed. The multimodal interaction front-end presented in [6], here integrated with the anger detection module, will be tested in the next future in order to validate both the accuracy of the approach in real conditions of use and the user acceptability and satisfaction. This will be done by means of both an objective and a subjective analysis. The former evaluation will be based on a background software module able to producing log-files containing all the details of the interaction session (time of interaction, number of touches on the pad, length of the speech utterance, etc.), in an evaluation release of the application the user will be requested a-posteriori to verify:

- if the ASR worked properly;
- if the request was correctly recognized and executed.

The analysis of the data collected in this way will be put in relation with those coming from a subjective investigation based on a questionnaire proposed to a further set of users (different from those involved in the former analysis) in order to estimate the subjective acceptability and the degree of satisfaction for the proposed application.

For what it concerns the data on which the Support Vector Machine classifier is trained, while we are currently using a corpus of acted emotions, we plan to use the recordings coming from the tests the system will undergo. We expect this will improve performance as the system will be retrained to work in final deployment conditions. The classifier will therefore be adapted to real-life conditions both in terms on spontaneous emotional display and in terms of recording environment as new recordings will include telephonic microphones quality and background noise.

Differently from what stated in [10], where the telephonic domain and the nature of the interaction did not encourage the introduction of an anger detection system in order to reduce the amount of hang-ups during dialogues, we believe that the mobile device domain will take advantage by the addition of an emotional state recognizer. In the case of apps for mobile devices requirements are different from those observed during telephonic dialogues and, provided that the Human-Computer Interface is well designed and correctly engineered, it is not really expected that the user closes the app before obtaining the required service. In this view, anger detection must be seen as a further effort made by the designer to convince users not to give up and close the app before reaching their goals.

6 Conclusions

We have presented a framework to design and implement multimodal interfaces with relatively little effort. As far as we know, anger detection and, in general, emotional feedback has not been taken into account in mobile applications before. The case study we presented shows a mobile application integrating speech recognition, anger detection and gesture analysis to implement a bus stops querying system. A basic release of the presented system, without speech and multimodal system is presently available on the Google Market (https://play.google.com/store/apps/details?id=it.unina.lab.citybusnapoli) and received excellent user reviews and more than 2600 downloads (April 2012), we consider this as a very effective usability test. Multimodal without emotive feedback is also being tested for usability by means of a subjective procedure, we are now undergoing formal testing of the complete system in order to verify its usability and its stability.

Acknowledgements

We would like to thank Vincenzo Galatà for providing the speech recordings from the yet unpublished multilingual emotional speech corpus €motion we used in our experiments. We would also like to thank Antonio Caso for assisting during the extension of the original framework to include the emotional module.

References

1. Ang, J., Dhillon, R., Krupski, A., Shriberg, E.and Stolcke, A.: Prosody-based automatic detection of annoyance and frustration in human-computer dialog. In: Proc. of of ICSLP. pp. 2037–2040 (2002)
2. Bodell, M., Dahl, D., Kliche, I., Larson, J., Tumuluri, R., Yudkowsky, M., Selvaraj, M., Porter, B., Raggett, D., Raman, T., Wahbe, A.: Multimodal architectures and interfaces (2011), http://www.w3.org/TR/mmi-arch/
3. Bolt, R.A.: "Put-that-there": Voice and gesture at the graphics interface. SIGGRAPH Comput. Graph. 14(3), 262–270 (1980)
4. Burkhardt, F., Polzehl, T., Stegmann, J., Metze, F., Huber, R.: Detecting real life anger. In: Proc. of ICASSP. pp. 4761–4764 (2009)
5. Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J., Young, R.M.: Four easy pieces for assessing the usability of multimodal interaction: the care properties. In: Proc. of INTERACT. pp. 115–120 (1995)
6. Cutugno, F., Leano, V.A., Mignini, G., Rinaldi, R.: Multimodal framework for mobile interaction. In: Proc. of AVI. pp. 197–203 (2012)
7. Dumas, B., Lalanne, D., Ingold, R.: Description languages for multimodal interaction: A set of guidelines and its illustration with SMUIML. Journal on Multimodal User Interfaces 3(3), 237–247 (2010)
8. Ehlen, P., Johnston, M.: Multimodal local search in speak4it. In: Proc. of IUI. pp. 435–436 (2011)
9. Galatà, V.: Production and perception of vocal emotions: a cross-linguistic and cross-cultural study, PhD Thesis - University of Calabria, Italy
10. Herm, O., Schmitt, A., Liscombe, J.: When calls go wrong: How to detect problematic calls based on log-files and emotions. In: Proc. of Interspeech. pp. 463–466 (2008)
11. Larson, J.A., Raman, T.V., Raggett, D., Bodell, M., Johnston, M., Kumar, S., Potter, S., Waters, K.: W3C multimodal interaction framework (2003), http://www.w3.org/TR/mmi-framework/
12. Petrillo, M., Cutugno, F.: A syllable segmentation algorithm for english and italian. In: Proc. of Eurospeech. pp. 2913–2916 (2003)
13. Petrushin, V.: Emotion in speech: Recognition and application to call centers. In: Proc. of ANNE [Online] (1999)

14. Polzehl, T., Schmitt, A., Metze, F.: Approaching multilingual emotion recognition from speech - on language dependency of acoustic/prosodic features for anger detection. In: Proc. of Speech Prosody (2010)
15. Yacoub, S., Simske, S., Lin, X., Burns, J.: Recognition of emotions in interactive voice response systems. In: Proc. of Eurospeech. pp. 729–732 (2003)
16. Zaykovskiy, D., Schmitt, A., Lutz, M.: New use of mobile phones: towards multimodal information access systems. In: Proc. of IE. pp. 255–259 (2007)

Human action recognition oriented to humanoid robots action reproduction

Stefano Michieletto and Emanuele Menegatti

Intelligent Autonomous Systems Lab (IAS-Lab)
Department of Information Engineering (DEI)
Faculty of Engineering, The University of Padua
Via Ognissanti 72, I-35129 Padova, Italy
`{stefano.michieletto,emg}@dei.unipd.it`
`http://robotics.dei.unipd.it`

Abstract. Our research aims at providing a humanoid robot with the ability of observing, learning, and reproducing actions performed by humans in order to acquire new skills. In other words, we want to apply artificial intelligence techniques to automatically recognize a human activity in order to make a humanoid robot able to reproduce it.This system has not only to distinguish between different actions, but also to represent them in a proper manner to allow a robot to reproduce the motion trajectories the demonstrator showed and learn new skills. Since the final system is going to be integrated in an autonomous humanoid robot (specifically model Aldebran Nao), we are working with an RGB-D sensor (Microsoft Kinect) that can be easily applied to it. This objective introduces also strict real-time constrains to the action recognition algorithm: we have opted for a probabilistic approach that offers good modeling and fast recognition performances.

Keywords: Action recognition, humanoid robots, programming by demonstration, imitation learning

1 Introduction

When the Turing Test was proposed, in 1950 [9], the idea was to compare humans and machines by looking at their responses. We would like to extend this criteria and compare humans and machines by looking at their actions. This regards the abilities to perceive the world, to learn from what happen around themselves, and the way they move with respect to the rest of the environment, that is the hallmark of an intelligent system. This project aims to provide a robot the capability to perform a task, recognizing actions from human demonstrators and learning from their movements.

A lot of work has been done from the 1950 in these fields, in particular in computer vision and robotics research. Computer vision goal is to extract information from a scene and structure it, and action recognition is assuming more and more importance in this field for its correlation with wide range of application such as biometrics, animation and interactive environments, video analysis and surveillance. The identification process can be divided into three stages: feature extraction, feature representation and action classification [10]. Two different approaches characterize features extraction: *model-based* and with *no-model* assumption. The first technique takes advantage of the prior knowledge of the body model to extract the selected features from the scene. The second one, instead, uses a common features extraction method. The features representation is strictly connected to the extraction phase. The *model-based* methods usually keep track of position and orientation of each part that composes the model. *No-model* methods, on the other hand, maintain the features characterize themselves: surface normals, RGB features, time-space information of each point and so on. The action classification can be approached by comparing static images with pre-stored samples (*template-based*[1]), by describing the system with a series of states with a certain probability and allowing switches between different states (*probability-based*[4][6]), or representing the motion with a sequence of grammatical rules (*grammar-based*[2]).

Robotics was defined as the science which studies the intelligent connection between perception and action[8] and mixes together mechanics, controls, computers and electronics, so that it benefits from advances in the different technologies. Recently, robots has been used in more and more areas such as *service robotics*, *field robotics*, or *human augmentation*. By the dawn of the new millennium, the "human" factor pushes robots to rapidly assume anthropomorphic appearance and the more robots interact with people safer, smarter and more independent they need to be, that why other disciplines were involved in the robotics area. The human ability to imitate behaviors is already present in even 12 - 21 day old infants [5]. Despite some hypothesis about the translation between visual and motion features[11], the mechanisms that express a demonstration to a set of internal motor primitives remain largely unknown. Several works in robotic imitation suggest that robots can perform new skills by learning key information from a demonstration [7]. More recently, this idea has been applied on humanoid robots as a natural way to teach how to perform a task in a dynamic environment [3].

The remainder of the paper is organized as follows: Section 2 describes the acquisition system. In Section 3 the features extraction method and their representation are introduced, while the action classification is described in Section 4. Conclusions and future works are contained in Section 5.

2 Acquisition system

The development of affordable RGB-D sensors is giving a rapid boost in computer vision research. These sensors are also reliable: they allow to natively capture RGB and depth information at good resolution and frame rate (the Microsoft Kinect takes 30 frames per second at 640x480 pixels of resolution). On the other hand, the sunlight plays havoc with the infrared pattern projected by the sensor and they cannot be used over eight meters. An indoor environment is not a real limitation for the humanoid robot we will use, so mounting RGB-D camera can be a very rich source of information for our work. The not invasive impact is another advantage of such sensors with respect to motion sensors and the robot can learn from every person in the scene.

In order to simplify the initial setting, the RGB-D sensor will not be immediately mounted on the robot. At first, the action recognition algorithm will work as a stand-alone system, hence we can assume that the data can be elaborated off-line. This assumption allows us to compare our algorithm with others at the state of the art, without the constrain of a camera moving on a robot. For this purpose, we are thinking of releasing a dataset of actions acquired as RGB-D data. Figure 1 shows some people performing actions for our preliminary tests. Once the algorithm will be efficient and robust enough, the robot movement will be considered and the system will be adjusted to prevent the performances from being affected by the robot movements.

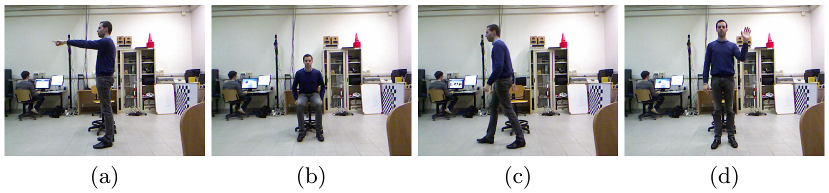


Fig. 1. Actions collected in our preliminary tests: pointing (a), sitting down(b), walking (c), and waving (d)

3 Features extraction and representation

A *model-based* approach will be adopted for features extraction. At the beginning of the project the OpenNI[1] freeware implementation of skeleton tracker will give us information about 25 skeletal joints covering the entire body. For each joint we will record position and orientation with respect to the camera frame. The `ROS::tf`[2] standard message will be used in order to exchange information between the acquisition process and the other algorithms we will implement. This choice is due to ROS[3] as common framework in this work. An example of the 3D model obtained from the tracker is showed in Figure 2. The OpenNI skeleton tracker is not open source and it will be probably replaced with a different one, in order to allow us to modify some internal parameters and improve the system.

[1] http://openni.org/
[2] http://www.ros.org/wiki/tf/
[3] http://www.ros.org/

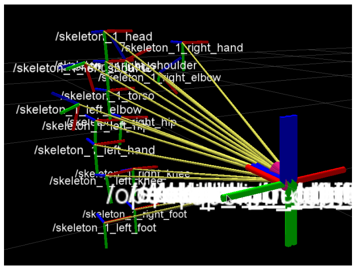


Fig. 2. 3D skeleton model (joints data obtained using OpenNI tracker). Each coordinates system (the RBG axis) represents position and rotation of a skeleton joint with respect to (yellow arrows) the camera frame.

4 Action classification

The action classification will lean on a *probability-based* model. Having a probabilistic model allows us to classify actions in a compact and efficient representation extracting the principal action constraints. This peculiarity is fundamental in working with actions involving a large number of degree of freedom, like in the human body. In fact, the demonstrations can vary one from each other and some movements could be not essential to reach the goal. A probabilistic approach makes also possible to perform incremental construction of a model, thus making easier to refine the representation with new demonstrations. The purpose of this work is make the robot able to reproduce the skills learned using the action recognition system. A regression process will be developed to archive an effective robot control with smooth trajectories in a fast and analytic way. The idea is to extract a "mean" trajectory that would change in the "variance" field of the acquired trajectories depending on the situation. Figure 3 shows the idea for an example trajectory in 2D space.

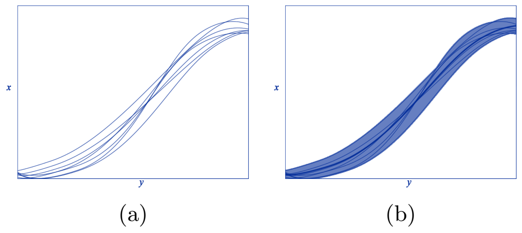


Fig. 3. Example of 2D trajectories extraction (a) and their regression (b).

5 Conclusion

In this paper we presented an human action recognition system designed to reproduce the acquired skills on a humanoid robot. We plan to test our system first in a simulated environment, and then with two different humanoids: VStone Robovie-X and Aldebaran Nao (Figure 4). Robovie-X is an affordable small robot perfect to test robustness and flexibility. Nao is bigger than Robovie-X and it can mount on it the RGB-D sensor to perform real-time tests on a moving robot.

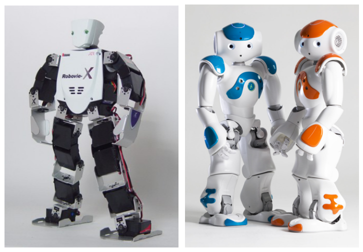


Fig. 4. Robot proposed for our tests: Robovie-X (a) and Nao (b).

References

1. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. Pattern Analysis and Machine Intelligence, IEEE Transactions on 23(3), 257–267 (Aug 2002)
2. Brand, M.: Understanding manipulation in video. In: Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (FG '96). pp. 94–. FG '96, IEEE Computer Society, Washington, DC, USA (1996)
3. Calinon, S.: Robot Programming by Demonstration: A Probabilistic Approach. EPFL/CRC Press (2009)
4. Gong, S., Xiang, T.: Recognition of group activities using dynamic probabilistic networks. In: Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2. pp. 742–. ICCV '03, IEEE Computer Society, Washington, DC, USA (2003)
5. Meltzoff, A.N., Moore, M.K.: Imitation of Facial and Manual Gestures by Human Neonates. Science 198(4312), 75–78 (Oct 1977)
6. Moënne-Loccoz, N., Brémond, F., Thonnat, M.: Recurrent bayesian network for the recognition of human behaviors from video. In: Proceedings of the 3rd international conference on Computer vision systems. pp. 68–77. ICVS'03, Springer-Verlag, Berlin, Heidelberg (2003)
7. Schaal, S.: Learning from demonstration. In: Advances in Neural Information Processing Systems 9 (1997)

8. Siciliano, B., Khatib, O. (eds.): Springer Handbook of Robotics. Springer (2008)
9. Turing, A.M.: Computing machinery and intelligence (1950)
10. Wei, W., Yunxiao, A.: Vision-based human motion recognition: A survey. In: Proceedings of the 2009 Second International Conference on Intelligent Networks and Intelligent Systems. pp. 386–389. ICINIS '09, IEEE Computer Society, Washington, DC, USA (2009)
11. Wolpert, D.M., Ghahramani, Z., Jordan, M.I.: An internal model for sensorimotor integration. Science 269, 1880–1882 (1995)

Semantics and Inference for Probabilistic Ontologies

Fabrizio Riguzzi, Elena Bellodi, Evelina Lamma, and Riccardo Zese

ENDIF – University of Ferrara, Italy,
email: {fabrizio.riguzzi, elena.bellodi, evelina.lamma}@unife.it, riccardo.zese@student.unife.it

Abstract. We present BUNDLE, a reasoner able to perform reasoning on probabilistic knowledge bases according to the semantics DISPONTE. In DISPONTE the axioms of a probabilistic ontology can be annotated with an epistemic or a statistical probability. The epistemic probability represents a degree of confidence in the axiom, while the statistical probability considers the populations to which the axiom is applied. BUNDLE exploits an underlying OWL DL reasoner, which is Pellet, that is able to return explanations for a query. However, it can work well with any reasoner able to return explanations for a query. The explanations are encoded in a Binary Decision Diagram from which the probability of the query is computed.

1 Introduction

Uncertainty has been recognized as an important feature for the Semantic Web [13]. In order to be able to represent and reason with probabilistic knowledge, various authors have advocated the use of probabilistic ontologies and many proposals have been put forward for allowing ontology languages, and OWL in particular, to represent uncertainty [10]. In the field of logic programming, the distribution semantics [11] has emerged as one of the most effective approaches.

In this paper we apply this approach to ontological languages and, in particular, to the OWL DL fragment, that is based on the description logic $\mathcal{SHOIN}(\mathbf{D})$. However, the approach is applicable in principle to any description logic. We called the approach DISPONTE for "DIstribution Semantics for Probabilistic ONTologiEs". The idea is to annotate axioms of a theory with a probability and assume that each axiom is independent of the others. As in Probabilistic Logic Programming, the probability of a query is computed from a covering set of explanations by solving a disjoint sum problem.

We also present the algorithm BUNDLE - for "Binary decision diagrams for Uncertain reasoNing on Description Logic thEories" - that performs inference over probabilistic ontologies.

The paper is organized as follows. Section 2 illustrates Description Logics. Section 3 presents DISPONTE. Section 4 describes BUNDLE and presents the results of the experimental comparison between the probabilistic reasoners BUNDLE and PRONTO [3].

2 Description Logics

Description Logics (DLs) are a family of formalisms used to represent knowledge. Studies on DLs are focused on finding ways to build intelligent applications, able to find the implicit consequences starting from the explicit knowledge. DLs are particularly useful for representing ontologies and have been adopted as the basis of the Semantic Web. For example, the OWL DL sublanguage of OWL is based on the $\mathcal{SHOIN}(\mathbf{D})$ DL. While DLs can be translated into predicate logic, they are usually represented using a syntax based on concepts and roles. A concept corresponds to a set of individuals of the domain while a role corresponds to a set of couples of individuals of the domain. In order to illustrate DLs, we now describe $\mathcal{SHOIN}$ following [6].

Let $\mathbf{A}$, $\mathbf{R}$ and $\mathbf{I}$ be sets of *atomic concepts*, *roles* and *individuals*, respectively. A *role* is either an atomic role $R \in \mathbf{R}$ or the inverse R^- of an atomic role $R \in \mathbf{R}$. We use $\mathbf{R}^-$ to denote the set of all inverses of roles in $\mathbf{R}$. A RBox $\mathcal{R}$ consists of a finite set of *transitivity axioms* ($Trans(R)$), where $R \in \mathbf{R}$, and *role inclusion axioms* ($R \sqsubseteq S$), where $R, S \in \mathbf{R} \cup \mathbf{R}^-$. *Concepts* are defined by induction as follows. Each $A \in \mathbf{A}$ is a concept, $\bot$ and $\top$ are concepts, and if $a \in \mathbf{I}$, then $\{a\}$ is a concept. If C, $C1$ and $C2$ are concepts and $R \in \mathbf{R} \cup \mathbf{R}^-$, then $(C_1 \sqcap C_2)$, $(C_1 \sqcup C_2)$, and $\neg C$ are concepts, as well as $\exists R.C$, $\forall R.C$, $n \geq R$ and $n \leq R$ for an integer $n \geq 0$. A *TBox* $\mathcal{T}$ is a finite set of *concept inclusion axioms* $C \sqsubseteq D$, where C and D are concepts. We use $C \equiv D$ to abbreviate $C \sqsubseteq D$ and $D \sqsubseteq C$. A *ABox* $\mathcal{A}$ is a finite set of *concept membership axioms* $a : C$, *role membership axioms* $(a, b) : R$, *equality axioms* $a = b$, and *inequality axioms* $a \neq b$, where C is a concept, $R \in \mathbf{R}$ and $a, b \in \mathbf{I}$. A *knowledge base* $\mathcal{K} = (\mathcal{T}, \mathcal{R}, \mathcal{A})$ consists of a TBox $\mathcal{T}$, an RBox $\mathcal{R}$ and an ABox $\mathcal{A}$.

The semantics of DLs can be given in a set-theoretic way or by transforming a DL knowledge base into a predicate logic theory and then using the model-theoretic semantics of the resulting theory.

$\mathcal{SHOIN}(\mathbf{D})$ adds to $\mathcal{SHOIN}$ datatype roles, i.e., roles that map an individual to an element of a datatype such as integers, floats, etc.

A query over a knowledge base is usually an axiom for which we want to test the entailment from the knowledge base.

3 The DISPONTE Semantics for Probabilistic Ontologies

A *probabilistic knowledge base* is a set of *certainly true axioms*, that take the form of DL axioms, of *epistemic probabilistic axioms* of the form $p ::_e E$ where p is a real number in $[0, 1]$ and E is a TBox, RBox or ABox axiom, and of *statistical probabilistic axioms* of the form $p ::_s E$ where p is a real number in $[0, 1]$ and E is a TBox or RBox axiom. In epistemic probabilistic axioms, p is interpreted as the degree of our belief in axiom E, while in statistical probabilistic axioms, p is interpreted as information regarding random individuals from certain populations. For example, an epistemic probabilistic concept inclusion axiom of the form $p ::_e C \sqsubseteq D$ represents the fact that we believe in the truth of $C \sqsubseteq D$ with probability p. A statistical probabilistic concept inclusion axiom of the form $p ::_s C \sqsubseteq D$ instead means that a random individual of class C has probability p of belonging to D, thus representing the statistical information that a fraction p of the individuals of C belong to D. In this way, the overlap between C and D is quantified. The difference between the two axioms is that, if two individuals belong to class C, the probability that they both belong to D according to the epistemic axiom is p, while according to the statistical axiom is $p \times p$.

In order to give a semantics to such probabilistic knowledge bases, we consider their translation into predicate logic. The idea of DISPONTE is to associate independent Boolean random variables to instantiations of the formula in predicate logic that is obtained from the translation of the axiom. An instantiation is a substitution θ for a logical formula F, consisting of pairs x/a, where x is a variable universally quantified by the outermost quantifier and a is an individual.

We now formally define the semantics in the following. An *atomic choice*, in this context, is a triple (F_i, θ_j, k), where F_i is the formula obtained by translating the ith axiom, θ_j is a substitution and $k \in \{0, 1\}$. A set of atomic choices κ is *consistent* if $(F_i, \theta_j, k) \in \kappa$ and $(F_i, \theta_j, m) \in \kappa$ implies $k = m$. A *composite choice* κ is a set of atomic choices and its probability is $P(\kappa) = \prod_{(F_i,\theta_j,1)\in\kappa} p_i \prod_{(F_i,\theta_j,0)\in\kappa}(1 - p_i)$. A *selection* σ is a total composite choice, i.e., an atomic choice for every instantiation of formulas of the theory. A selection σ identifies a theory w_σ called a *world* in this way: $w_\sigma = \{F_i\theta_j | (F_i, \theta_j, 1) \in \sigma\}$. A composite choice κ identifies a set of worlds $\omega_\kappa = \{w_\sigma | \sigma \in \mathcal{S}_T, \sigma \supseteq \kappa\}$, where $\mathcal{S}_T$ is the set of all selections. A composite choice κ is an *explanation* for a query Q if Q is entailed by every world of ω_κ. We define the set of worlds identified by a set of composite choices K as $\omega_K = \bigcup_{\kappa\in K} \omega_\kappa$. A set of composite choices K is *covering* with respect to Q if every world w_σ in which Q is entailed is such that $w_\sigma \in \omega_K$. Two composite choices κ_1 and κ_2 are *incompatible* if their union is inconsistent. A set K of composite choices is *mutually incompatible* if for all $\kappa_1 \in K, \kappa_2 \in K, \kappa_1 \neq \kappa_2 \Rightarrow \kappa_1$ and κ_2 are incompatible.

Now we can define a unique probability measure $\mu : \Omega \to [0, 1]$, where $\Omega = \{\omega_K | K \text{ is a finite set of finite composite choices}\}$. μ is defined by $\mu(\omega_K) = \sum_{\kappa\in K'} P(\kappa)$ where K' is a finite mutually incompatible set of finite composite choices such that $\omega_K = \omega_{K'}$.

Example 1. Let us consider the following simple ontology, inspired by the `people+pets` ontology proposed in [8]:

$$\begin{aligned}
&\exists hasAnimal.Pet \sqsubseteq PetOwner\\
&(kevin, fluffy) : hasAnimal\\
&(kevin, tom) : hasAnimal\\
&fluffy : Cat\\
&tom : Cat\\
0.6 ::_e\ &Cat \sqsubseteq Pet
\end{aligned}$$

The predicate logic formula (without external quantifiers) equivalent to the only probabilistic axiom above is: $F_1 = Cat(x) \to Pet(x)$. A covering set of explanations for the query axiom $Q = kevin : PetOwner$ is $K = \{\kappa_1\}$ with $\kappa_1 = \{(F_1, \emptyset, 1)\}$. In fact, there is only one probabilistic axiom and its presence is necessary to entail the query. This is also a mutually exclusive set of explanations since it contains a single composite choice so $P(Q) = 0.6$.

Example 2. If the axiom $0.6 ::_e Cat \sqsubseteq Pet$ in Example 1 is replaced by $0.6 ::_s Cat \sqsubseteq Pet$ then the query would have the explanations $K = \{\kappa_1, \kappa_2\}$, where $\kappa_1 = \{(F_1, \{x/fluffy\}, 1)\}$ and $\kappa_2 = \{(F_1, \{x/tom\}, 1)\}$. The set $K' = \{\kappa'_1, \kappa'_2\}$, where

$\kappa'_1 = \{(F_1, \{x/\mathit{fluffy}\}, 1)\}$ and $\kappa'_2 = \{(F_1, \{x/\mathit{fluffy}\}, 0), (F_1, \{x/tom\}, 1)\}$, is such that $\omega_K = \omega_{K'}$ and K' is mutually incompatible, so $P(Q) = 0.6 + 0.6 \cdot 0.4 = 0.84$. K' can be found by applying the splitting algorithm of [9].

4 Query Answering and Experiments

The BUNDLE algorithm computes the probability of queries from a probabilistic ontology that follows the DISPONTE semantics. BUNDLE needs an underlying DL reasoner, such as Pellet [12], that is able to return explanations for queries. BUNDLE builds a Binary Decision Diagram (BDD) from the set of explanations. The BDD is then used to compute the probability using the dynamic programming algorithm of [1].

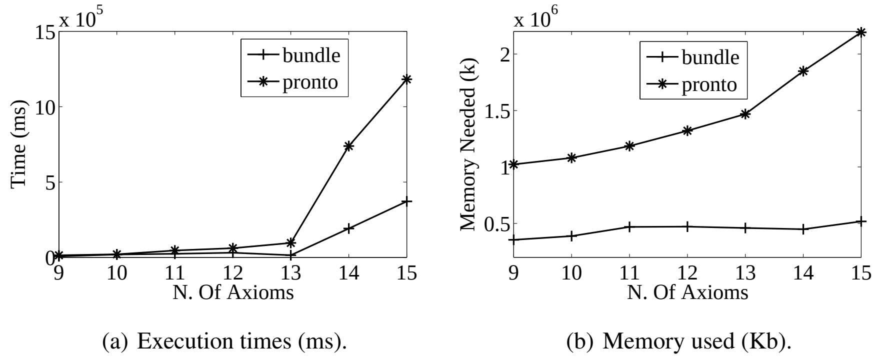
If the knowledge base contains only epistemic probabilistic axioms, Pellet can be used directly as the underlying ontology reasoner. If the knowledge base contains also statistical probabilistic axioms, Pellet needs to be modified so that it records, besides the axioms that have been used to answer the query, also the individuals to which they are applied. Each tableau expansion rule used by Pellet returns a set of uninstantiated axioms. Therefore we have modified Pellet's expansion rules in order to return a set of couples $(axiom, substitution)$ instead of simple axioms.

In order to evaluate the performances of BUNDLE, we followed the methodology of [4] where the system PRONTO [3] is used to answer queries to increasingly complex ontologies, obtained by randomly sampling axioms from a large probabilistic ontology for breast cancer risk assesment (BRCA). This ontology is divided into two parts: a classical and a probabilistic part. The probabilistic part contains conditional constraints [2] of the form $(D|C)[l, u]$ that informally mean "generally, if an object belongs to C, then it belongs to D with a probability in $[l, u]$". For instance, the statement that an average woman has up to 12.3% of developing breast cancer in her lifetime is expressed by

$$(WomanUnderAbsoluteBRCRisk|Woman)[0, 0.123]$$

Tests have been defined by randomly sampling a subset of conditional constraints from the probabilistic part and adding these constraints to the classical part to form a random ontology. We varied the number of constraints from 9 to 15, and, for each number, we repeatedly sampled ontologies and tested them for consistency. We stopped sampling when we obtained 100 consistent ontologies for each number of constraints. The ontologies have then been translated into DISPONTE by replacing the constraint $(D|C)[l, u]$ with the axiom $l ::_s C \sqsubseteq D$. For each ontology we performed the query $a : C$, where a is a new individual for which a number of class assertions are added to the ontology: a was randomly assigned to each class appearing in the sampled conditional constraints with probability 0.6. Class C of the query was randomly selected among those representing women under increased and lifetime risk.

We then applied both BUNDLE and PRONTO to each generated test and we measured the execution time and the memory used. Figure 1(a) shows the average execution time as a function of the number of axioms and, similarly, Figure 1(b) shows the average amount of memory used. These graphs show that BUNDLE is faster and uses less memory than PRONTO. The comparison is not meant to be interpreted in terms of a superiority of BUNDLE, since the two systems treat ontologies with different semantics, rather, it should provide a qualitative evaluation of the complexity of the DISPONTE semantics with respect to the one of [2] that is based on lexicographic entailment [5] and Nilsson's probabilistic logic [7]. In the future we plan to investigate the application of BUNDLE to other real life ontologies.



(a) Execution times (ms).

(b) Memory used (Kb).

Fig. 1. Comparison between BUNDLE and PRONTO.

References

1. De Raedt, L., Kimmig, A., Toivonen, H.: ProbLog: A probabilistic Prolog and its application in link discovery. In: International Joint Conference on Artificial Intelligence. pp. 2462–2467 (2007)
2. Giugno, R., Lukasiewicz, T.: P-SHOQ(D): A probabilistic extension of SHOQ(D) for probabilistic ontologies in the semantic web. In: European Conference on Logics in Artificial Intelligence. LNCS, vol. 2424, pp. 86–97. Springer (2002)
3. Klinov, P.: Pronto: A non-monotonic probabilistic description logic reasoner. In: European Semantic Web Conference. LNCS, vol. 5021, pp. 822–826. Springer (2008)
4. Klinov, P., Parsia, B.: Optimization and evaluation of reasoning in probabilistic description logic: Towards a systematic approach. In: International Semantic Web Conference. LNCS, vol. 5318, pp. 213–228. Springer (2008)
5. Lehmann, D.J.: Another perspective on default reasoning. Ann. Math. Artif. Intell. 15(1), 61–82 (1995)
6. Lukasiewicz, T., Straccia, U.: Managing uncertainty and vagueness in description logics for the semantic web. Journal of Web Semantics 6(4), 291–308 (2008)
7. Nilsson, N.J.: Probabilistic logic. Artificial Intelligence 28(1), 71–87 (1986)
8. Patel-Schneider, P, F., Horrocks, I., Bechhofer, S.: Tutorial on OWL (2003), http://www.cs.man.ac.uk/ horrocks/ISWC2003/Tutorial/
9. Poole, D.: Abducing through negation as failure: stable models within the independent choice logic. Journal of Logic Programming 44(1-3), 5–35 (2000)
10. Predoiu, L., Stuckenschmidt, H.: Probabilistic models for the semantic web: A survey. In: The Semantic Web for Knowledge and Data Management: Technologies and Practices. IGI Global (2008)

11. Sato, T.: A statistical learning method for logic programs with distribution semantics. In: International Conference on Logic Programming. pp. 715–729. MIT Press (1995)
12. Sirin, E., Parsia, B., Cuenca-Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics 5(2), 51–53 (2007)
13. URW3-XG: Uncertainty reasoning for the World Wide Web, final report (2005)

OTTHO: An Artificial Player for a Complex Language Game

Giovanni Semeraro, Pasquale Lops, Marco de Gemmis, and Pierpaolo Basile

Dept. of Computer Science - University of Bari
Via E. Orabona, 4 - 70125 Bari - Italia
{semeraro,lops,degemmis,basilepp}@di.uniba.it

Abstract. This paper describes OTTHO (On the Tip of my THOught), an artificial player able to solve a very popular language game, called "The Guillotine", broadcast by the Italian National TV company. The game demands knowledge covering a broad range of topics, such as politics, literature, history, proverbs, and popular culture. The rule of the game is simple: the player observes five words, generally unrelated to each other, and in one minute she has to provide a sixth word, semantically connected to the others. In order to find the solution, a human being has to perform a complex memory retrieval task within the facts retained in her own knowledge, concerning the meanings of thousands of words and their contextual relations. In order to make this task executable by machines, machine reading techniques are exploited for knowledge extraction from the web, while Artificial Intelligence techniques are used to infer new knowledge, in the form of keywords, from the extracted information.

1 Background and Motivation

The literature classifies games related to the language in two main categories: *word games* and *language games* [5]. *Word games* do not involve true language, because word meanings are not important. An example of word game is *Scrabble*, in which players take turn placing letters in a grid to form words. *Language games*, such as crosswords or "Who wants to be a millionaire?", strongly involve natural language, since word meanings play an important role. Language games draw their challenge and excitement from the richness and ambiguity of natural language, which is, on the other side, the main source of complexity for machines. OTTHO is a system designed to provide solutions for *The Guillotine* game, in which the player is given a set of five words (*clues*), each linked in some way to a specific word that represents the *unique* solution of the game. Words are unrelated to each other, but each of them is strongly related to the solution. Once the five clues are given, the player has *one minute* to guess the right answer. For example, given the clues *sin, newton, doctor, pie, new york*, the solution is *apple*, because in the popular Christian tradition the apple is the forbidden fruit in the Book of Genesis, that is the symbol of the original *sin*, *Newton* discovered the gravity by means of an apple, "an apple a day takes the *doctor* away" is a proverb, the apple *pie* is a fruit cake, and *new york* city is called "the big apple". Often the solution is not so intuitive and only players with a strong background knowledge are able to find the correct word. Indeed, in order to find the solution, a human being has to perform a complex *memory retrieval* task within the facts retained in her own knowledge, concerning the meanings of thousands of words and their contextual relations [9].

No fixed sets of rules are sufficient to define the game play, thus solving the game depends exclusively on the background knowledge of the system, which is created by *machine reading* techniques. They analyze *unstructured* information stored in open knowledge sources on the Web, such as dictionaries and Wikipedia, and build a memory of linguistic competencies and world facts that can be effectively exploited by the system for a deeper understanding of clues. Relatedness between terms, providing the evidence of a strong relationship between words, is the key factor for finding a set of candidate words that likely contains the solution. To this purpose, OTTHO exploits a reasoning mechanism based on Spreading Activation techniques [4, 3] which allows matching clues with the background knowledge. The main motivation for designing an artificial player for this game is the challenge of providing the machine with both the cultural and linguistic background knowledge which makes it similar to a human being, with the ability of interpreting natural language documents and reasoning on their content. Our feeling is that the approach presented in this work has a great potential for other more practical applications besides solving a language game, which are mentioned in the last section of the paper.

2 System Description

An extended knowledge base must be built for representing the *cultural* and *linguistic* background knowledge of the artificial player. After a deep analysis of the correlation between the clues and the solution, the following knowledge sources have been processed to build the memory of the system:

- **Encyclopedia** – the Italian version of Wikipedia;
- **Dictionary** – the De Mauro Paravia Italian on-line dictionary (no longer available);
- **Compound forms** – groups of words that often go together having a specific meaning, crawled from the IntraText Digital Library (`http://www.intratext.com/bsi/listapolirematiche/indalfa.htm`) and the on-line dictionary TLIO - Tesoro della Lingua Italiana delle Origini (`http://ovipc44.ovi.cnr.it/Tliopoli/`);
- **Proverbs and Aphorisms** – the Italian version of Wikiquote;
- **Movies** – descriptions of Italian movies crawled from the Internet Movie Database (`http://www.imdb.com`);
- **Songs** – Italian songs crawled from OnlyLyrics (`http://www.onlylyrics.com/`);
- **Books** – book titles crawled from several web sites.

The above mentioned types of sources have different characteristics, therefore an important problem is to define a uniform representation of the information they store, which is discussed in the next section.

2.1 The Memory of the System: Cognitive Units

Formerly we modeled each source as a term-term matrix whose cells represent the degree of correlation between the term on the row and the one on the column, according to specific heuristics [8]. In the new version of the system, we adopt a novel strategy based on the ACT theory of fact memory [1], according to which information in long term memory of human beings is encoded as *Cognitive Units* (CUs) that form an interconnected network. A cognitive unit is a piece of information (e.g. a proposition) we can hold consciously in our focus of attention, together with all the links (many of which are unconscious) that can be established with other parts of our cognitive structure. According to this idea, we see knowledge sources as repositories of CUs.

Because of the heterogeneity of the knowledge sources involved in the process, two problems must be solved in the implementation of the step that turns knowledge sources into a unique *machine-readable* knowledge base, with concepts represented in a homogeneous format:

- identification of the basic unit representing a concept in each specific knowledge source (e.g. a Wikipedia article, a lemma in a dictionary);
- definition of a unique representation model for cognitive units in the background knowledge.

Since the information provided by the knowledge sources is represented in textual form, we regard a CU as the structured textual description of a concept. For each knowledge source included in the memory of OTTHO, the basic unit describing a concept is chosen: a *lemma* in the Dictionary, an *article* in Wikipedia, etc. Basic units are turned into CUs by machine reading techniques which analyze the text and build the corresponding descriptions of recognized concepts. Each CU is represented by two fields:

1. **head** – words identifying the concept represented by the CU;
2. **body** – words describing the concept.

In a more compact way:

```
CU = [head|body]
```

For example, the Wikipedia article that provides the description of the concept *Artificial Intelligence*[1] is turned into the corresponding CU and stored in a repository of cognitive units:

[1] http://en.wikipedia.org/wiki/Artificial_intelligence

```
CU_124 = [Artificial Intelligence (3.56) |AI (1.23),
          machine (1.14), computer science (2.58),
          Alan Turing (2.77)]
```

Keywords in CUs are assigned a weight representing how important they are for that concept, similar to the bag of words approach in Information Retrieval (IR). The main advantage of this representation strategy is that an IR model can be adopted for retrieving relevant CUs associated with a keyword. 743,192 CUs have been defined: 584,527 for the Encyclopedia, 126,741 for the Dictionary, 10,744 for Compound Forms, 11,257 for Proverbs and Aphorisms, and 9,923 for Songs, Movies and Books. The complete description of the machine reading process can be found in [7]. This kind of "knowledge infusion" into the system creates a memory of world facts and linguistic knowledge.

As depicted in Figure 1, *clues* are used to query the memory of OTTHO, i.e. the CU repository, in order to retrieve the most appropriate "pieces of knowledge" related to them. Both clues and retrieved CUs are then passed to the reasoning algorithm, which produces a *new* list of keywords associated with them, which are possible solutions of the game. The reasoning mechanism is described in the following section.

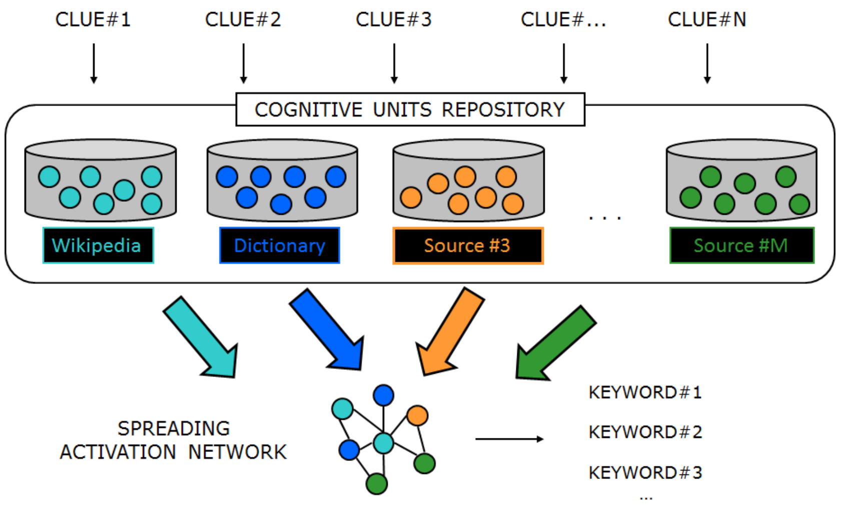


Fig. 1. Process for finding candidate solutions for a run of the game

2.2 The Brain of the System: Spreading Activation Network

Spreading activation has proved to be a model with a high degree of explanatory power in cognitive psychology [1]. One of the merits of this model is that it captures the way knowledge is represented as well as the way it is processed. In this model, knowledge is represented in terms of nodes and associative pathways between the nodes. Specifically, concepts are represented in memory as nodes, and relations between concepts as associative pathways between the corresponding nodes. When part of the memory network is activated, activation spreads along the associative pathways to related areas in memory. The spread of activation mechanism selects the areas of the memory network that are more ready to further cognitive processing.

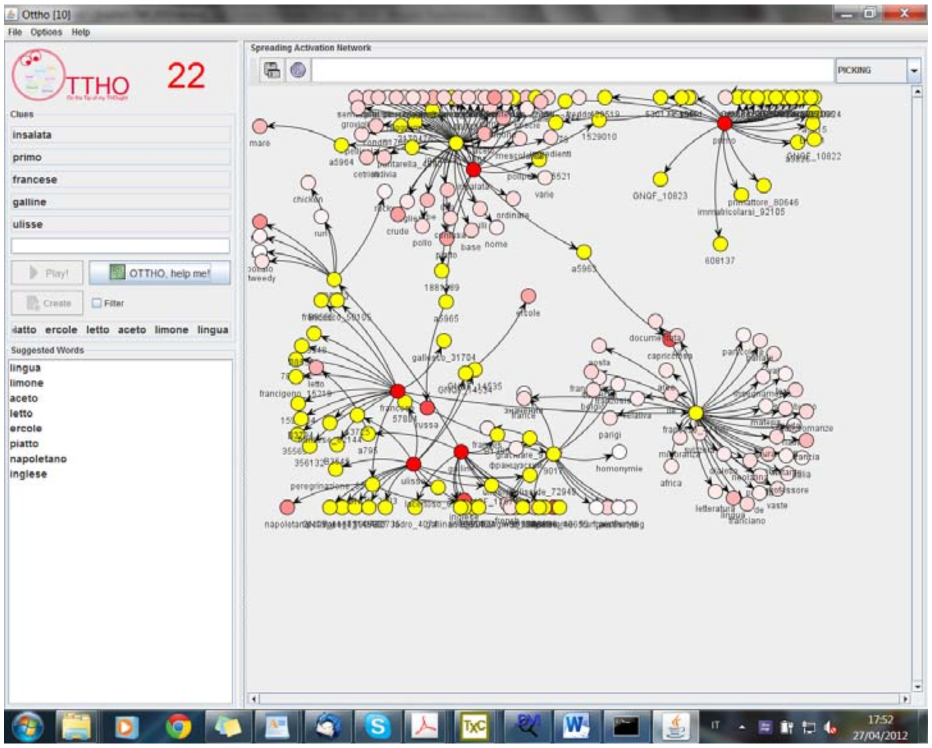
Since words and their meanings are stored in the mind in a network-like structure [1], we adopted this model as the reasoning mechanism of OTTHO. In the network for The Guillotine game, called SAN - Spreading Activation Network, nodes represent either words or CUs, while links denote associations between them, obtained from CU repositories. Links are weighted with a score that measures the strength of the association.

The SAN for a run of the game is built in 3 steps: (1) nodes corresponding to clues are included in the SAN; (2) clues are connected to the most appropriate CUs retrieved by a search mechanism which queries the CU repositories by using clues; (3) retrieved CUs are connected to nodes representing the most informative terms associated with them. An example of SAN is depicted in Figure 2. All in all, the SAN is the part of the background knowledge of the system which is related to the clues of the game to be solved. The spreading process over the SAN starts from clue nodes and propagates first to CUs and then to words connected to CUs. In fact, it is a search process which selects, among all the nodes in the SAN, those which are strongly connected to clues, and therefore are good candidate solutions. Technical details about the spreading algorithm are reported in [7].

3 Into and Beyond the Game

Figure 2 shows the OTTHO user interface. In this scenario, the system supports the human player by showing, within the text area on the bottom-left, some candidate solutions for the clues visualized on the top-left of the window. A timer is displayed on the clues, near the OTTHO logo, which warns the player on time to provide the answer. The SAN is depicted on the right. The solution for this run is *letto (bed)*, which actually appears in the list of suggestions.

Figure 3 emphasizes the part of the SAN in which the solution is found. Notice that the solution is connected with the clue “galline” since the idiom “andare a *letto* con le *galline*” (“to go to bed with the chickens”, that means “very early”) was found in the CU repository. By clicking on the CU node of the idiom *a795* and then on the “information” button on the top of the SAN, the explanation for the solution is shown by OTTHO in the “polirematiche” box on the left.



Clues
insalata
primo
francese
galline
ulisse

piatto ercole letto aceto limone lingua

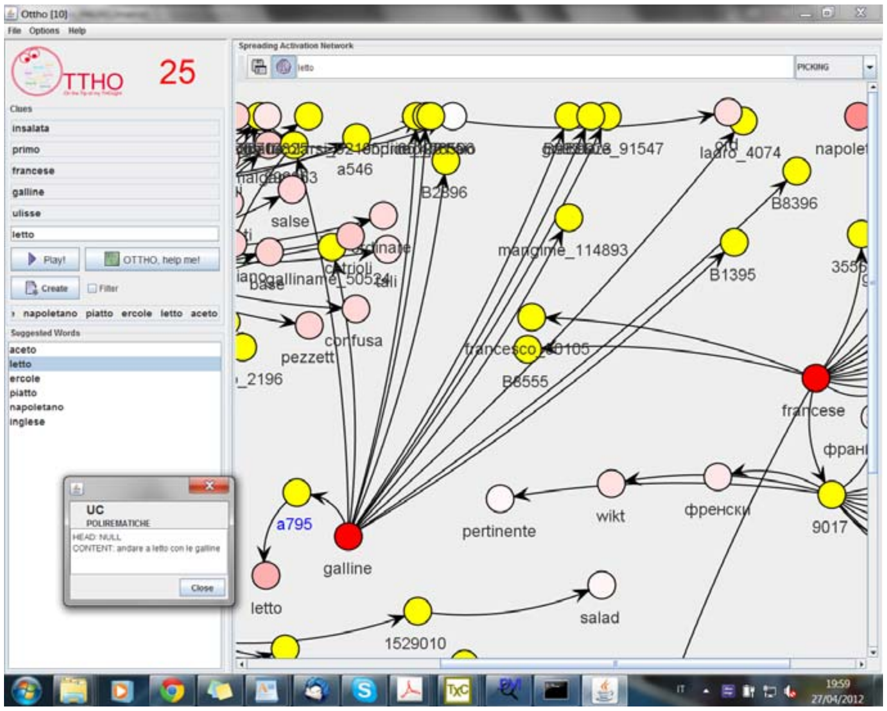
Suggested Words
lingua
limone
aceto
letto
ercole
piatto
napoletano
inglese

Fig. 2. A run of the game

The system, besides supporting the player, could also assist authors of the game for the verification of the uniqueness of the "official" solution. Indeed, the "create" button (on the left of the GUI) allows users to propose their own clues and to verify whether other words, in addition to the official answer, are consistent with the clues. In conclusion, the proposed system has a great potential for other practical applications both in the area of Information Retrieval and Information Filtering. For example, words suggested by OTTHO could be used by search engines for intelligent query expansion [2] or by content-based recommender systems for the computation of similarity between items [6].

References

1. Anderson, J.R.: A Spreading Activation Theory of Memory. Journal of Verbal Learning and Verbal Behavior 22, 261–295 (1983)
2. Carpineto, C., Romano, G.: A survey of automatic query expansion in information retrieval. ACM Comput. Surv. 44(1), 1 (2012)
3. Collins, A.M., Loftus, E.F.: A Spreading Activation Theory of Semantic Processing. Psychological Review 82(6), 407–428 (1975)



UC
POLIREMATICHE
HEAD: NULL
CONTENT: andare a letto con le galline

Fig. 3. Solution found in the SAN with explanation by OTTHO

4. Crestani, F.: Application of Spreading Activation Techniques in Information Retrieval. Artificial Intelligence 11(6), 453–482 (1997)
5. Littman, M.L.: Review: Computer language games. In: Revised Papers from the Second International Conference on Computers and Games. pp. 396–404. CG '00, Springer-Verlag, London, UK (2002)
6. Lops, P., de Gemmis, M., Semeraro, G.: Content-based Recommender Systems: State of the Art and Trends. In: Recommender Systems Handbook, pp. 73–105. Springer (2011)
7. Semeraro, G., de Gemmis, M., Lops, P., Basile, P.: Knowledge Infusion from Open Knowledge Sources: an Artificial Player for a Language Game. IEEE Intelligent Systems DOI: http://doi.ieeecomputersociety.org/10.1109/MIS.2011.37. In press
8. Semeraro, G., Lops, P., Basile, P., de Gemmis, M.: On the Tip of My Thought: Playing the Guillotine Game. In: Proc. of the 21st Int. Joint Conference on Artificial Intelligence. pp. 1543–1548. Morgan Kaufmann (2009)
9. Spitzer, M.: The Mind within the Net: Models of Learning, Thinking, and Acting. MIT Press (2000)

Towards an Interactive Personal Care System driven by Sensor Data

Stefano Bragaglia, Paola Mello, and Davide Sottara

DEIS, Facoltà di Ingegneria, Università di Bologna
Viale Risorgimento 2, 40136 Bologna (BO) Italy
stefano.bragaglia@unibo.it, paola.mello@unibo.it,
davide.sottara2@unibo.it

Abstract. This demo presents an integrated application, exploiting technologies derived from various branches of Artificial Intelligence. The prototype, when triggered by an appropriate event, interacts with a person and expects them to perform a simple acknowledgement gesture: the following actions, then, depend on the actual recognition of this gesture. In order to achieve this goal, a combination of data stream processing and rule based programming is used. While the more quantitative components exploit techniques such as Clustering or (Hidden) Markov Models, the higher levels, more qualitative and declarative, are based on well known frameworks such as Event Calculus and Fuzzy Logic.

Keywords: Complex Event Processing, Event Calculus, Sensor Data Fusion, Activity Recognition

1 Introduction

Intelligent beings need senses to interact with the environment they live in: the same principle holds for artificially intelligent entities when they are applied outside of purely virtual contexts, in the real world. Much effort is being spent in emulating smell (processing signals acquired through electronic noses), taste (using electronic tongues) and touch, although hearing and especially sight are arguably the subject of an even larger amount of research and application development. Focusing on computer vision, observing the position or the shape of people and objects is extremely relevant when dealing with problems such as pathfinding, tracking, planning or monitoring. Moreover, analysing the visual information is often the prelude to a decisional process, where actions are scheduled depending on the sensed inputs and the current goals. In our case study, part of the DEPICT project, we are addressing the monitoring of an elderly person during their daily activity, recognizing, if possible, the insurgence of undesired short and long term conditions, such as frailty, and ultimately trying to prevent severely detrimental events such as falls. Should a fall or similar event actually happen, however, it is essential to recognize it as soon as possible, in order to take the appropriate actions. To this end, we are planning to use a mobile platform equipped with optical and audio sensors and a decisional processing unit: the device would track or locate the person as needed, using the sensors to acquire information on their current status and capabilities. In this paper, however, we will not discuss the mobile platform, but focus on the sensors it is equipped with, and the collected information. This information will be analyzed to identify and isolate, in ascending order of abstraction, *poses*, *gestures*, *actions* and *activities* [1]. A pose is a specific position assumed by one or more parts of the body, such as "sitting", "standing" or "arms folded"; a gesture is a simple, atomic movement of a specific body part (e.g. "waving hand", "turning head"); actions are more complex interactions such as "walking" or "standing up", while activities are composite actions, usually performed with a goal in mind. Recognizing patterns directly at each level of abstraction is a relevant research task of its own, but it may also be necessary to share and correlate information between the levels. From a monitoring perspective, the more general information may provide the necessary context for a proper analysis of the more detailed one: for example, one might be interested in a postural analysis of the spine when a person is walking or standing up from a chair, but not when a person is sleeping.

2 AI Techniques

In order to analyse a person's movements in a flexible and scalable way, we propose the adoption of a hybrid architecture, combining low level image and signal processing techniques with a more high level reasoning system. The core of the system is developed using the "Knowledge Integration Platform" Drools[1], an open source suite which is particulary suitable for this kind of applications. It is based on a production rule engine, allowing to encode knowledge in the form of `if-then` rules. A rule's premise is normally a logic, declarative construction stating the conditions for the application of some consequent action; the consequences, instead, are more operational and define which actions should be executed (including the generation of new data) when a premise is satisfied. Drools, then, builds its additional capabilities on top of its core engine: it supports, among other things, temporal reasoning, a limited form of functional programming and reasoning under uncertainty and/or vagueness [6]. Being object oriented and written in Java, it is platform independent and can be easily integrated with other existing components. Using this platform, we built our demo application implementing and integrating other well known techniques.

Vision sub-system. At the data input level, we used a Kinect hardware sensor due to its robustness, availability and low price. It combines a traditional camera with an infrared depth camera, allowing to reconstruct both 2D and 3D images. We used it in combination with the open source OpenNI[2] middleware, in particular exploiting its tracking component, which allows to identify and trace the position of humanoid figures in a scene. For each figure, the coordinates of their "joints" (neck, elbow, hip, etc. . .) are estimated and sampled with a frequency of 30Hz.

[1] http://www.jboss.org/drools
[2] http://www.OpenNI.org

Vocal sub-system. We rely on the open source projects FreeTTS[3] and Sphinx-4[4] for speech synthesis and recognition, respectively.

Semantic domain model. Ontologies are formal descriptions of a domain, defining the relevant concepts and the relationships between them. An ontology is readable by domain experts, but can also be processed by a (semantic) reasoner to infer and make additional knowledge explicit, check the logical consistency of a set of statements and recognize (classify) individual entities given their properties. For our specific case, we are developing a simple ontology of the body parts (joints) and a second ontology of poses and acts. The ontologies are then converted into an object model [5], composed of appropriate classes and interfaces, which can be used to model facts and write rules.

Event Calculus. The Event Calculus [2] is another well known formalism, used to represent the effects of actions and changes on a domain. The base formulation of the EC consists of a small set of simple logical axioms, which correlate the happening of *events* with *fluents*. An event is a relevant state change in a monitored domain, taking place at a specific point in time; fluents, instead, denote relevant domain properties and the time intervals during which they hold. From a more reactive perspective, the fluents define the overall state of a system through its relevant properties; the events, instead, mark the transitions between those states.

Complex Event Processing. In addition to flipping fluents, events may become relevant because of their relation to other events: tipical relations include causality, temporal sequencing or aggregation [4]. Causality indicates that an event is a direct consequence of another; sequencing imposes constraints on the order events may appear in; aggregation allows to define higher level, more abstract events from a set of simpler ones. Exploiting these relations is important as the number and frequency of events increases, in order to limit the amount of information, filtering and preserving only what is relevant.

Fuzzy Logic. When complex or context-specific conditions are involved, it may be difficult to define when a property is definitely true or false. Instead, it may be easier to provide a vague definition, allowing the property to hold up to some intermediate degree, defined on a scale of values. Fuzzy logic [3] deals with this kind of graduality, extending traditional logic to support formulas involving graded predicates. In this kind of logic, "linguistic" predicates such as `old` or `tall` can replace crisp, quantitative constraints with vague (and smoother) qualitative expressions. Likewise, logical formulas evaluate to a degree rather than being either valid or not.

3 Demo Outline.

The techniques listed in the previous section have been used as building blocks for a simple demo application, demonstrating a user/machine interaction, based on vision and supported by the exchange of some vocal messages. The abstract

[3] http://freetts.sourceforge.net/docs/index.php

[4] http://cmusphinx.sourceforge.net/sphinx4/

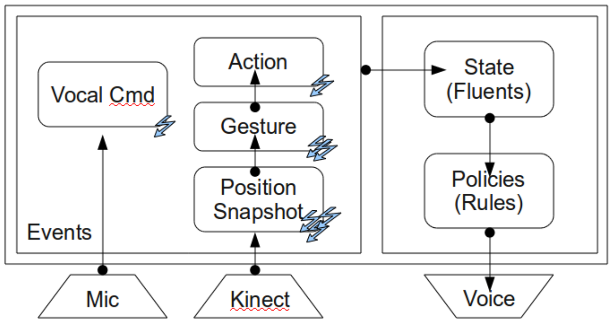


Fig. 1. Architectural outline

system architecture is depicted in Figure 1. The simple protocol we propose as a use case, instead, is a simplified interaction pattern where a monitoring system is checking that a person has at least some degree of control over their cognitive and physical capabilities. This is especially important when monitoring elderly patients, who might be suffering from (progressively) debilitating conditions such as Parkinson's disease and require constant checks. Similarly, in case of falls, estimating the person's level of consciousness may be extremely important to determine the best emergency rescue plan.

To this end, for each person tracked by the Kinect sensor, we generate an object model (defined in the ontology) describing its skeleton with its joints and their position in space. The actual coordinates are averaged over a window of 10 samples to reduce noise: every time a new average is computed, an event is generated to notify the updated coordinates. The coordinates are then matched to predetermined patterns in order to detect specific poses: each snapshot provides a vector of 63 features (3 coordinates and 1 certainty factor for each one of the 15 joints, plus the 3 coordinates of the center of mass) which can be used for the classification. The definition of a pose may involve one or more joints, up to the whole skeleton and can be expressed in several ways. In this work, we adopted a "semantic" [1] approach, providing definitions in the form of constraints (rules) on the joints' coordinates. To be more robust, we used fuzzy predicates (e.g. `leftHand.Y is high`) rather than numeric thresholds, but the infrastructure could easily support other types of classifiers (e.g. neural networks or clusterers). Regardless of the classification technique, we assume the state of being in a given pose can be modelled using a fuzzy fluent, i.e. a fluent with a gradual degree of truth in the range $[0, 1]$. A pose, in fact, is usually mantained for a limited amount of time and, during that period, the actual body position may not fit the definition perfectly. In particular, we consider the maximum degree of compatibility (similarity) between the sampled positions and the pose definition over the minimal interval when the compatibility is greater than 0 (i.e. it is not impossible that the body is assuming the considered pose). Technically, we keep a fluent for each person and pose, declipping it when the compatibility between its joints and the pose is strictly greater than 0 and clipping it as soon as it becomes 0. Given the poses and their validity intervals and degrees, denoted by the fluents, it is possible to define gestures and actions as sequences of poses and/or gestures. While we are planning to consider techniques such as (semi-)Hidden Markov Models, currently we continue adopting a "semantic", CEP-like approach at all levels of abstraction. Gestures and actions, then, are defined using rules involving lower level fluents. Their recognition is enabled only if they are relevant given the current context: the event sequencing rules, in fact, are conditioned by other state fluents, whose state in turn depends by other events generated by the environment.

Usage. The user is expected to launch the application and wait until the tracking system has completed its calibration phase, fact which is notified by a vocal message. From this point on, until the user leaves the camera scope, the position of their joints in a 3D space will be continuously estimated. When the application enters a special recognition state, triggered pressing a button or uttering the word "*Help*", the user has 60 seconds to execute a specific sequence of minor actions, in particular raising their left and right hand in a sequence or stating "*Fine*". With this example, we simulate a possible alert condition, where the system believes that something bad might have happened to the person and wants some feedback on their health and their interaction capabilities.

If the user responds with gestures, the system will observe the vertical coordinates of the hands: a hand will be raised to a degree, which will be the higher the more the hand will be above the shoulder level. A hand partially lifted will still allow to reach the goal, but the final result will be less than optimal. At any given time, the recognition process considers the highest point a hand has reached so far, unless the hand is lowered below the shoulder (the score is reset to 0). If both hands have been completey lifted, the user has completed their task and the alert condition is cancelled; otherwise, the system will check the current score when the timeout expires. In the worst case scenario, at least one of the hands has not been raised at all, leading to a complete failure; finally, in intermediate situations, at least one hand will not have been lifted completely, leading to a partial score. The outcome will be indicated using a color code: green for complete success, red for complete failure and shades of orange for the intermediate cases, in addition to a vocal response message. If the user answers vocally, instead, the alarm will be cleared and no further action will be taken.

4 Conclusions

We have shown an example of a tightly integrated, leveraging both quantitative and qualitative AI-based tools. Despite its simplicity, it proves the feasibility of an interactive, intelligent care system with sensor fusion and decision support capabilities.

Acknowledgments

This work has been funded by the DEIS project DEPICT.

References

1. J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Comput. Surv.*, 43(3):16:1–16:43, April 2011.
2. F. Chesani, P. Mello, M. Montali, and P. Torroni. A logic-based, reactive calculus of events. *Fundamenta Informaticae*, 105(1):135–161, 2010.
3. Petr Hájek. *Metamathematics of Fuzzy Logic*, volume 4 of *Trends in Logic: Studia Logica Library*. Kluwer Academic Publishers, Dordrecht, 1998.
4. David C. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.
5. G. Meditskos and N. Bassiliades. A rule-based object-oriented OWL reasoner. *IEEE Transactions on Knowledge and Data Engineering*, 20(3):397–410, 2008.
6. D. Sottara, P. Mello, and M. Proctor. A configurable rete-oo engine for reasoning with different types of imperfect information. *IEEE Trans. Knowl. Data Eng.*, 22(11):1535–1548, 2010.

4D-Particle filter localization for a simulated UAV

Anna Chiara Bellini

annachiara.bellini@gmail.com

Abstract. Particle filters are a mathematical method that can be used to build a belief about the location of a robot in an environment that has a few known features, called landmarks. Each particle is a virtual copy of the robot, representing a possible robot's location, that moves in the environment as the robot does, and with an associated probability, representing the likelihood of each location. Through cycles of move robot-update particles, sense-get likelihood and resample, the probability distribution changes to reflect the current belief. The environment is divided in cells and the probability of being in each cell is the sum of the particles in that cell. The orientation is the weighted mean of the orientations of the particles in a cell. The demo uses the simulator built for the AIRobots EU project.

Keywords: Robot localization, Particle filters, Probabilistic localization, UAV

1 Introduction

Autonomous robots that move in an environment need to have a very precise knowledge about their own location in order to make correct decision about the next action or motion. While usually some information about the general location is available, it is most often the case that the error is too large to be sufficient for the robot, but the robot can use information from its sensors to refine this information up to the degree of precision that it needs.

For instance, a car on a road has a rough idea of its location thanks to a GPS and a map of the road, but the error of a normal car GPS is in the order of +/- 3-5 meters, which is not sufficient for a car to drive autonomously. Similarly, a robot inspecting a plant knows the map of the plant and might be initialized to a known starting location, but, as it proceeds around the plant, the accrued error gets too large to allow the robot to move in narrow corridors or pass doors. It may be the case that the initial conditions are unknown, making it impossible to rely on knowledge of the previous path to determine the location.

In each of these cases, there are distinguishable features that the robot can perceive, and whose location on the map is known, such as a tree that the car can see at the side of the road and that it can individuate on a satellite picture of the area, or a lamp on the plant's wall that is documented on the electrical project. Such distinguishable and known features are called landmarks, and are what the robot can use to refine its knowledge. However, the landmarks are often indistinguishable from one another, i.e. there can be several trees by the road, and probably all of the lamps in a plant are identical. Similarly, the information of the map could be incomplete, like a lamp could be off, or a tree might have been planted after the satellite picture was taken. Particle filters allow robots to localize themselves in these difficult situations.

The work described in this paper was done as an academic project within the AI Fundamentals course held by Prof. Paola Mello at the Faculty of Engineering in Bologna, using also material from an online course held by Prof. Sebastian Thrun at the Udacity website [4]. Techniques learnt in the two courses were applied in the simulation environment built for the AIRobots EU project [3], where the author has developed the 3D simulation part.

2 Particle filters

When information is so partial and uncertain and the environment grows in size and complexity, deterministic or analytical methods for localization are bound to fail, because the number of possible cases to be considered becomes too large just after a few percept-action cycles. Particle filters [1][2][5] represent the current belief of the robot as a discrete probability distribution over the entire environment, or at least over the portion of the environment that is being considered, such as the area individuated by GPS. In the middle of each percept-action cycle, after the perception and before deciding on the action, an update-resample step is taken and a new belief is built based only on the current belief state, the percept from the sensors (i.e. landmark sensing) and the previous action, so the computational needs don't grow over time.

2.1 Multimodal belief

In an environment, there could be two locations A and B that are very similar with respect to the landmarks, and the best assumption that the robot can make is that it is either close to location A or to location B. This is true especially at the beginning of localization, when the robot has very little knowledge, so the probability distribution that represents the belief must be multimodal, i.e. have multiple possible peaks. Over the subsequent update-resample cycles this belief could become unimodal, but even when the best assumption is to be in one of a few locations, this information could be sufficient for the robot to take correct actions.

2.2 Particles

The way that this belief is represented is by making a large number of hypotheses about the possible location of the robot, and each of these hypotheses is called a particle. A particle acts as a virtual copy of the robot, and perceives and moves just as the robot does, but it does so on the map of the environment, not in the real one. The more the particles' perceptions agree with the robot's ones, the higher the probability that the particle is close to the robot's location. After each update step, some particles survive or die according to their relative probability, but the total number of particles considered is fixed. The belief of the robot is given by the density of the particles: the more particles are in a specific area of the map, the higher the probability that the robot is in that area. Motion and perception on the map are computationally very easy, involving just simple arithmetical operations like addition, multiplication and roots, allowing the use of very large number of particles and precise localization.

Figure 1 shows a sample 2D world where a robot localizes itself according to a colour sensor, where both motion and sensing are affected by a Gaussian error. The colour of the border is related to the number of particles in the cell.

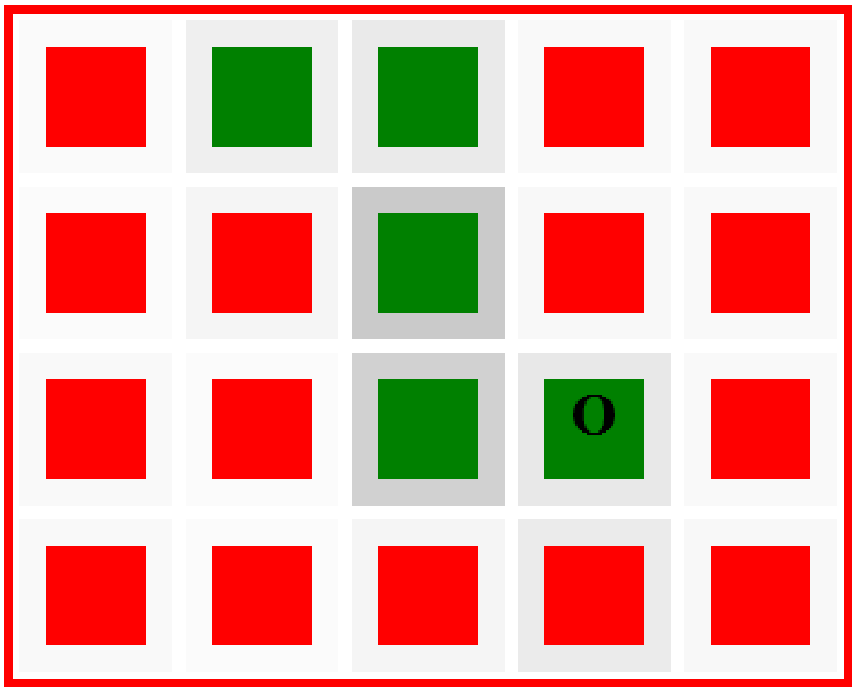


Figure 1 A sample 2D World, border darkness shows belief after some sense-update steps

2.3 The algorithm [4]

Initial distribution.

When the particle filter is initialized, whatever knowledge is available can be used to distribute the particles in the environment. In the most general case, particles are distributed at random over the entire environment.

```
N = <number_of_particles>
particles = []
```

```
for i in range(N):
  particles.append(createRandomParticle())
```

The update step.

Updating a particle means moving it according to the robot's motion:

```
m = <robot motion>
for i in range(N):
    particles[i].move(m)
```

and calculating its likelihood according to the current robot's measurements:

```
p = <robot percepts>
w = [] # likelihood of each particle
for i in range(N):
    w[i].append(particles[i].getLikelihood(p))
```

Resampling.

This is the crucial step of the particle filters, where a whole new set of particles is built:

```
new_particles = []
for i in range(N):
    sampled_part = pick_particle_with_prob (particles, w)
    new_particles.append(sampled_part.copy())
particles = new_particles
```

Determining the most likely location.

The belief of the robot about it's own location is given by the number of particles in each world cell:

```
world_cells = <portions of the environment>
for cell in world_cells:
   cell.probability = count_particles(particles, cell)
```

Taking errors into account.

The key feature that makes the particle filter work is uncertainty: the real robot has many uncertainties, due to motion, sensors' error, noise, external influences. The sensor's error is used in computing the `parti-cle.getLikelihood(measurement)` function, and allows for giving some likelihood even to particles whose virtual measurements differ from the robot's own measurement. For instance, if the robot senses a landmark at 1 meter, it could be at 0.95, or 1.05, and particles in that range should be considered quite likely. But it could also be that the sensor is having a momentarily failure, so even particles at 2 meters from the landmark should be given some residual probability.

Error in the particle's motion update step not only reflect the uncertainty in the real robot's motion, but also help spread the particles over a local area: in the resampling step, many copies of a single original particle can be made, and they will all be at the orginal particle's location. But when they move with a Gaussian error, they will spread over a local area, creating new possible locations.

3 The case study: AIRobots

The demonstration is based on the simulator that was built for the AIRobots project [3]. This case study was chosen because the author is the development team of the simulator, and wanted to explore an alternative method of localizing the robot. The robot is an UAV, an autonomous flying robot built to inspect plants. When the robot is inside the plant, maybe inspecting a boiler or a chimney, it is out of sight and needs precise localization to accomplish its task. The map of the environment is known, and landmarks information can be perceived using artificial vision, distance sensors, lasers or other. The robot's location and orientation is actually made of six coordinates: x, y, z, yaw, pitch and roll, but pitch and roll can easily be read from the onboard sensors. Also, the robot stays horizontal, deviating from this pose only slightly and only for moving, therefore there are only four coordinates to be determined by localization.

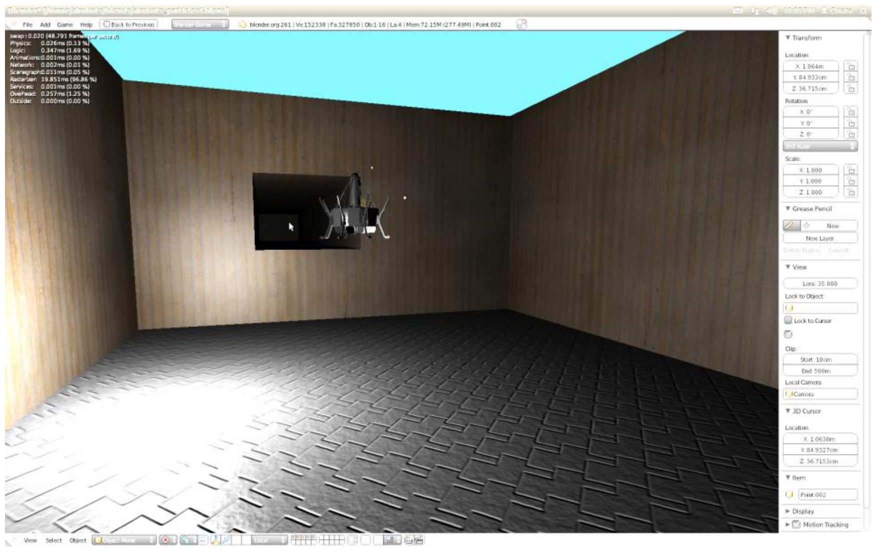
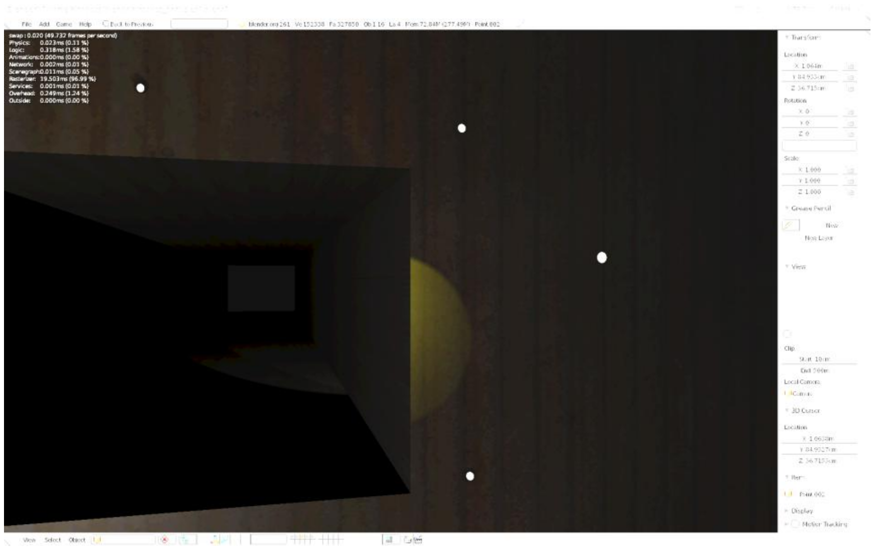


Figure 2 A dim simulation environment where the robot has little information

For this demonstration, the robot moves in a multiple-room environment, where the rooms have different sizes. The initial location of the robot is random, i.e. the robot doesn't have any clue to its relative position. Using a single low-res camera, the robot is able to individuate bright points, corresponding to lamps placed in the environment. Figure 3 shows a sample onboard view where bright dots are the lamps.



```
swap : 0.020 (49.732 frames per second)
Physics:     0.023ms (0.11 %)
Logic:       0.318ms (1.58 %)
Animations:0.000ms (0.00 %)
Network:     0.002ms (0.01 %)
Scenegraph:0.011ms (0.05 %)
Rasterizer:  19.503ms (96.99 %)
Services:    0.001ms (0.01 %)
Overhead:    0.249ms (1.24 %)
Outside:     0.000ms (0.00 %)
```

Figure 3 Bright indicate the landmarks that can be seen from the onboard camera.

The demonstration shows how the robot moves in the "real" 3D environment and how its belief evolves as it moves, showing that from very simple information, like the points of the lamps in the camera image, the robot can infer all of the four coordinates.

4 Conclusions

Particle filters are a probabilistic method that can be used to localize a robot in a known environment, when very precise localization is necessary. The method is very easy to implement, with limited memory and computation demands, and works even when the information is scarce, like simple bright points on a camera image to infer four spatial coordinates.

5 References

1. S. Thrun; W. Burgard; D. Fox;, "Probabilistic robotic". MIT press 2006. Chapter 4.

2. S.J. Russell; P. Norvig;, "Artificial Intelligence - a modern approach. Third edition." Prentice Hall 2011. Chapters 13-15.
3. AIRobots project website: http://www.airobots.eu/. In particular Deliverable D4.2 for the simulator.
4. S. Thrun. CS 373 – Building a Robotic Car – Online course at: http://www.udacity.com/
5. F. Dellaert; F. Fox; W. Burgard; S. Thrun; "Monte Carlo localization for mobile robots". Proceedings from the IEEE international conference on Robotics and Automation. 1999
6. Turgut, B.; Martin, R.P.; , "Restarting Particle Filters: An Approach to Improve the Performance of Dynamic Indoor Localization," Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE , vol., no., pp.1-7, Nov. 30 2009-Dec. 4 2009
7. Sangjin Hong; Jinseok Lee; Athalye, A.; Djuric, P.M.; We-Duke Cho; , "Design Methodology for Domain Specific Parameterizable Particle Filter Realizations," Circuits and Systems I: Regular Papers, IEEE Transactions on , vol.54, no.9, pp.1987-2000, Sept. 2007

Behind the scenes of Sudoku: Application of genetic algorithms for the optimal fun.

Thomas Bridi

Thomas.bridi@studio.unibo.it

Abstract. This work discusses about algorithms belonging to the branch of artificial intelligence for the generation of Sudoku puzzle. It will be demonstrated how the use of algorithms related to the constraint programming and genetic algorithms can improve the generation of puzzles in order to make the game more competitive and therefore more attractive to the public. In particular, it will be used an algorithm similar to the forward checking for the generation of a population of deterministic puzzles with a feasible solution and then will be used a genetic algorithm to evolve the population in order to optimize a function that rates the difficulty of their resolution.

Keywords: genetic algorithms, constraint programming, optimal solution, Sudoku generation.

1 Introduction

In this work we study the case of using artificial intelligence techniques for the generation of Sudoku, in particular genetic algorithms; we will see how these techniques can be used in areas such as journals specialized in this game to try to improve the player experience.

Chapter 2 will get a brief overview of the Sudoku game, its characteristics and the difference between probabilistic and deterministic Sudoku.

In chapter 3 we will study how the problem on generating Sudoku can be mapped as constraint satisfaction problem and we will see the limit of this kind of approach.

Chapter 4 gives a brief overview of genetic algorithms and how they are used for finding optimal solutions.

In chapter 5 we will see some related work about Sudoku and genetic algorithms.

In chapter 6 we will study the problem of generating more difficult Sudoku with the aids of genetic algorithms.

In chapter 7 we will make considerations about the results obtained from the algorithm proposed above.

2 The Game of Sudoku

As well known the Sudoku is a logical game. First of all we have to introduce some technical definitions: we will call cell the box where we are going to insert a number, grid the set of 81 cells, placed in 9x9, that makes up the Sudoku; sub-grid is a set of 3x3 cells, then we have nine rows and nine columns. The games starts with a partially filled grid and the target is to fill the remaining cells with digits from 1 to 9, constraints are that each number must appear only once in each row, column and sub-grid.

We can have two kind of Sudoku: deterministic and probabilistic. Probabilistic Sudoku is when a starting grid can be filled in different ways and more than one way reach to the objective (i.e. an empty grid is a probabilistic Sudoku). From now on we will skip the case of probabilistic Sudoku for investigating further issues related to deterministic Sudoku.

A deterministic Sudoku consists of an initial grid, partially filled, which admit a unique solution; this is a big advantage for the player that at each step will always have a valid move to do, unlike the probabilistic version that in some steps can admit more possible moves that could lead to several different solutions or to an invalid grid.

3 Generating a Sudoku using Constraints Satisfaction Problems

As we have seen previously in the definition of Sudoku we deal with constraints, then it seems logical to map the problem as a constraints satisfaction problem and it works perfectly. This is an example of the algorithm for the puzzle generation in pseudo language:

```
given G an empty grid;
do
  update the constraints of each cell;
  C := random selected cell from G;
  PossibleValues := values satisfying the constraints of C;
  V := random element from PossibleValues;
  set V as value of C;
while G does not admit solution;
```

Where “update the constraints of each cell” and “G does not admit solution” consist to apply the constraints described by the rules of the game. In order to make a more attractive puzzle, we must make it more difficult to solve; to do this we have to modify the “G does not admit solution” function taking advantage of more sophisticated techniques to detect more complex Sudoku.

In our case we do not need backtracking because this algorithm is designed to work on a large amount of puzzles, the use of backtracking is not necessary and also goes to degrade the performance in terms of memory usage, so in case of failure the algorithm will restart from an empty grid; for simplicity we will call this ad-hoc algorithm Naïve Forward Checking because it use the forward checking constraint propagation but not the backtracking.

Is intuitive to understand that in this way we will not have the security of generating more complex Sudoku but only to recognize it, then the next step is to understand how it is possible to get closer to the optimal solution of our problem, thanks to artificial intelligence.

4 An introduction to Genetics Algorithms

Genetics Algorithm is a part of Evolutionary Computation. The principle under genetic algorithms is the same principle of the evolutionary theory of Darwin: in an environment, individuals that survive and reproduce, are those with better characteristics, so their genes are partially passed on to the next generation. In evolutionary computation an individual is a possible solution of a problem, a population is a set of individuals, there is a fitness function that describe how many solutions are near to the optimal solution of the problem, every individual is obtained starting from a genotype, a genotype is the representation of all the variables that can make an individual different from another.

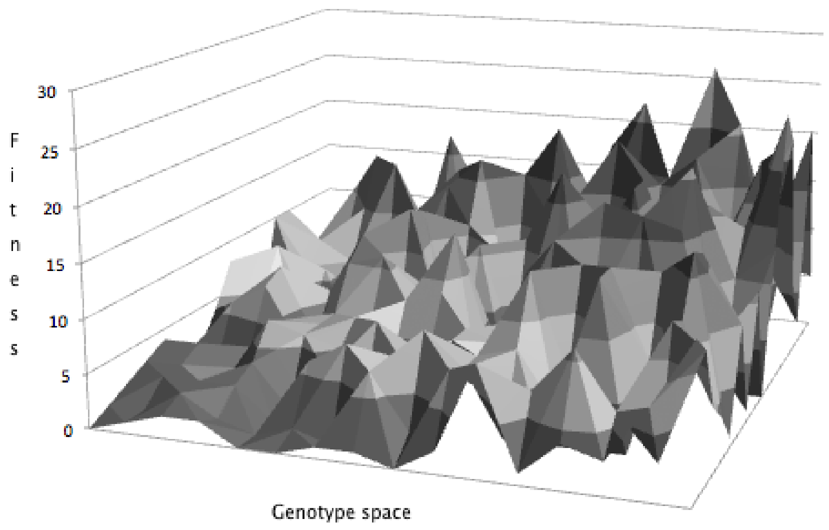


Fig. 1. Example of a fitness function of genotype space

We have a set of operations that makes a population evolve:

- **Recombination**: create a new individual from two parents.
- **Mutation**: evolve an individual modifying its genotype.

- **Insertion**: select old and new individuals to make a new population.

All these operations are done on selected individuals based on their fitness function and then theoretically every generation should get closer and closer to the optimal solution of the problem.

5 Related work

It is not easy to find words “Sudoku” and “Genetic algorithms” together in the literature especially for the generation of puzzles, M. Gold in 2005 published an application that use the approach of genetic algorithms to solve a Sudoku. In 2006 T.Mantere and J.Koljonen published a paper that describe how genetic algorithms can solve a puzzle used like a genome and also by applying swap operations; if given a blank puzzle, the algorithm fills up the grid generating a valid and solved Sudoku, but that is unplayable, the next step would be to remove the numbers from the grid to get a Sudoku with a feasible solution and the fewest number.

The approach used by T. Mantere and J. Kolijonen is completely different from the approach adopted in this work, the approach proposed for the puzzle generation use the algorithm proposed in chapter 3 for the generation of every single Sudoku but it uses a genetic algorithm for the heuristics, in particular the heuristics is about choosing to fill a mutated grid by selecting the cells in the same order used for the original grid, but using the value obtained by the mutation where possible.

6 Genetics Algorithm applied to the Sudoku generation problem

We have seen how genetic algorithms work to obtain the optimal solution. Our aim is not to get the most difficult Sudoku (undecidable problem), but to ensure that the generation of Sudoku meet a certain constraint of difficulty, common problem for anyone involved in the publication of Sudoku in magazines and games, therefore operations like recombination but especially mutation are perfectly suited to our problem.

First of all we have to decide how to set up our fitness function; as mentioned above we have several Sudoku solving techniques, we show the used techniques in the experiment in order of difficulty:

- Pencil mark
- Naked Single
- Hidden Single

Our fitness function is made to give to the puzzle a difficulty based to how many times is used a determinate techniques; every technique has a weight: Pencil mark 0, Naked Single 1 and Hidden Single 3; so a puzzle that require 2 Pencil Mark, 32 Naked Single and 7 Hidden Single moves will have difficulty 51.

A deterministic puzzle with a feasible solution is an individual of our population; its genotype is made by the list of random selected cells and their values (Chapter 3); at every step of the genetic algorithm we are going to select randomly individual on the basis of fitness and then will be applied a genotype mutation on it; finally, the mutated individuals will be reintroduced into the population.

7 Conclusions

Lastly we have made a prototype that creates first a random population of Sudoku puzzle using a Naive Forward Checking algorithm, than we have created the genotype of any individual by the filled cells and its value. The selected individuals by their fitness value has been mutated in some value in the genotype then the puzzle regenerated.

This algorithm will be repeated for a certain number of times and every times the average difficulty of the population is recalculated.

The result was that by applying the mutation, the average difficulty of the population tends asymptotically to a certain value with some small deviation from time to time; this asymptote depends on the fitness function but also by the number of techniques used by the solving algorithm.

Another test was made to calculate the percentage of successfully mutated grids and the percentage of mutation that produced a better solution, this test was done on starting population of about 40200 Sudoku grids and the result was that in average only the 8,24% of starting grid ported successfully to a deterministic puzzle before the fail of the naive forward checking algorithm; a possible way to increase this percentage could be to implement backtracking at least for the puzzle generation from genotype; the second result was that the percentage of mutation leading to a more difficult Sudoku is 35,88%.

About performance: this algorithm takes 3 hours for generating and mutating 10000 puzzle.

8 References

1. Andrea Roli: An introduction to evolutionary computation - http://lia.deis.unibo.it/Courses/AI/fundamentalsAI2011-12/lucidi/seminari/roli/intro-evolutionary_computation2012.pdf
2. John R. Koza, Riccardo Poli: A Genetic programming tutorial - http://lia.deis.unibo.it/Courses/AI/fundamentalsAI2009-10/lucidi/seminari/roli/gptutorial.pdf
3. Peter Norvig, Stuart J. Russell: Intelligenza Artificiale. Un approccio moderno.
4. E. Delucchi, G. Gaiffi, L. Pernazza: Passatempi e giochi: alla ricerca di problem e soluzioni - http://www.dm.unipi.it/~gaiffi/papers/giochi.pdf
5. Mathias Weller: Counting, Generating and Solving Sudoku - http://theinf1.informatik.uni-jena.de/publications/sudoku-weller08.pdf
6. Silvano Martello: Ricerca Operativa per la Laurea Magistrale

ProgramD_TV![*]

Una chatbot estesa

Emiliano Caponi, Irene Fabrini, Andrea Vanzo

Ingegneria Informatica Laurea Magistrale Università di Tor Vergata[1]

Abstract. Le Chatbots sono applicazioni che simulano il dialogo intelligente con esseri umani.

Il paradigma di realizzazione più utilizzato è l'approccio Stimolo-Risposta (S/R). Lo Stimolo è un insieme di frasi attese dalla chatbot, la Risposta sono le frasi restituite all'utente. Spesso tale approccio limita le potenzialità dell'interazione: la chatbot risponde solo quando la frase sottomessa dall'utente corrisponde ad una delle frasi attese. L'obiettivo del lavoro è il "rilassamento" di S/R attraverso due approcci: la semantica relazionale (relazione semantica tra parole) e la semantica distribuzionale (correlazione statistica tra parole). Attraverso tali meccanismi, la chatbot può rispondere anche in presenza di forte similarità tra la frase di input ed una frase attesa, mantenendo ancora coerenti le risposte. L'adozione di questi approcci permette, quindi, di migliorare le prestazioni base della chatbot.

Keywords: NLP, dialogo, semantica

1 Introduzione

Tra le applicazioni più diffuse dell'elaborazione del linguaggio naturale ci sono l'estrazione e la manipolazione di informazioni.

Tra le altre, ci siamo interessati all'analisi e sviluppo delle Chatbots che hanno il compito di simulare il dialogo e l'interazione tra un essere umano ed un agente. Rendere più efficaci tali sistemi ne permetterebbe l'inserimento in applicazioni industriali di diverse tipologie. Uno dei paradigmi più utilizzati per realizzare una Chatbot è lo Stimolo-Risposta (S/R). Lo Stimolo è rappresentato dall'insieme delle frasi che sono attese dalla chatbot, mentre la Risposta è l'insieme delle frasi che la chatbot ritorna al partner della comunicazione. Negli anni si sono sviluppati diversi linguaggi e piattaforme utili allo sviluppo delle chatbot. Le tecnologie prese in esame in questo lavoro sono l'AIML[2] (Artificial Intelligence Markup Language) e ProgramD[3]. L'AIML si presenta come estensione del linguaggio XML e la sua

[*] Estensione di ProgramD da parte di studenti di Tor Vergata (TV)

[1] Corso di Intelligenza Artificiale tenuto dalla Prof. M.T. Pazienza, definizione del tema progettuale da parte del prof. F. M. Zanzotto, tutoraggio del dot. A. Turbati

[2] http://www.alicebot.org/TR/2001/WD-aiml-1.0.1-20011018-005.html

[3] http://aitools.org/Program_D

caratteristica è proprio la codifica del paradigma (S/R). Gli stimoli sono elementi Pattern, mentre le risposte corrispondono ad elementi Template. ProgramD è la piattaforma utilizzata nel lavoro che permette l'implementazione degli standard AIML attraverso il linguaggio Java. Talvolta il paradigma S/R utilizzato da AIML-ProgramD può essere limitante nell'interazione tra utente e chatbot a causa della rigidità del paradigma stesso: la chatbot risponde solo nel caso di perfetta corrispondenza (matching) tra la frase di input ed uno degli elementi Pattern AIML. Data la variabilità lessicale dei linguaggi naturali, un concetto è espresso attraverso diverse forme frasali, che la chatbot non sempre è in grado di riconoscere. Per superare tale limitazione, si è cercato di identificare un approccio alternativo che è stato oggetto della nostra attività progettuale.

L'idea di base è perciò quella di sostituire il semplice pattern-matching con un approccio basato sulla valutazione della similarità tra termini presenti nello stimolo e quelli presenti nella risposta. In questo modo, il successo di una interrogazione non sarà dipendente dalle forme superficiali, ma guidato dal valore di similarità.

2 Metodologie utilizzate

L'obiettivo generale del lavoro è quello di estendere ProgramD attraverso l'utilizzo sia della semantica su dizionari strutturati che della semantica distribuzionale con l'intento di estendere il paradigma S/R implementato da ProgramD.

2.1 Semantica relazionale su dizionari strutturati: EuroWordNet

Nella semantica relazionale: due parole w_1 e w_2 sono correlate semanticamente se sono in una qualsiasi relazione tra: iperonimia, sinonimia, antonimia... Tali relazioni sono implementate in risorse quali dizionari strutturati o thesauri. Nel progetto si utilizza EuroWordNet[4] per l'italiano che è un database multilingue con reti di parole (wordnet) per alcune lingue europee. Gli EuroWordNet sono strutturati nello stesso modo del WordNet per l'inglese (Princeton WordNet)[5].

2.2 Semantica distribuzionale

La semantica distribuzionale considera come due parole w_1 e w_2 co-occorrono all'interno della stessa frase o di una finestra prestabilita in fase di progetto, considerando la loro distribuzione. In particolare, si sfrutta la Latent Semantic Analysis (LSA) che permette di estrarre e rappresentare il significato delle parole statisticamente, da un vasto insieme di documenti. LSA analizza le relazioni tra un insieme di documenti e i termini che essi contengono producendo un insieme di concetti (o topic) relativi ai documenti e ai termini. LSA assume che parole che sono

[4] http://www.ilc.cnr.it/viewpage.php/sez=ricerca/id=820/vers=ita

[5] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, «Introduction to WordNet: An On-line Lexical Database,» *International Journal Lexicography*, pp. 235-244, 1990.

vicine di significato occorreranno vicine nel testo (co-occorrenza). La matrice di partenza è una matrice termini per documenti, estratta da un corpus. Attraverso una trasformazione matriciale (SVD – Singular Value Decomposition, produce una approssimazione della matrice di partenza con rango minore), si estraggono nuove dimensioni (topic) che catturano meglio il contenuto informativo della matrice di partenza.

3 Descrizione ProgramD_TV

Gli elementi essenziali di ProgramD sono:

1) Core: modulo che permette di configurare tutti gli elementi per l'esecuzione della ChatBot;
2) Parser: effettua l'analisi dei file AIML ed XML;
3) Graph: contiene elementi quali Nodemapper e Graphmaster che hanno il compito di implementare il pattern S/R;
4) Multiplexor: ha il compito di gestire i flussi di input e di output;
5) Shell: l'interfaccia grafica che permette l'interazione Utente/Chatbot.

Nella figura 1 si può vedere una schematizzazione della struttura associata a ProgramD.

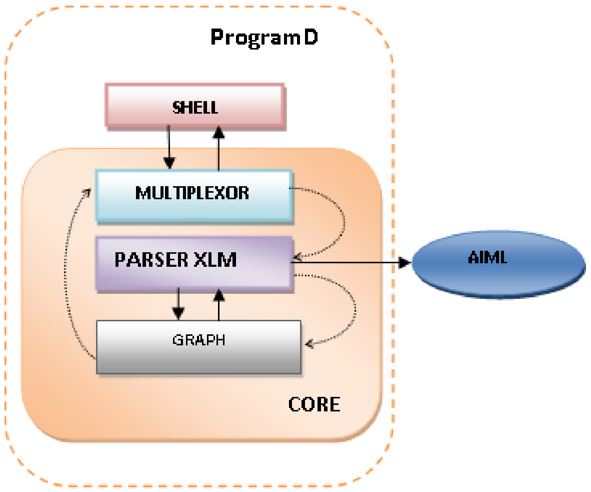


Fig. 1. Overview architettura ProgramD

Il modulo preso in esame è il Graph ed in particolare l'elemento GraphMaster, che ha il compito di costruire l'albero delle frasi attese dalla Chatbot e gestire il meccanismo di matching responsabile della navigazione. La costruzione dell'albero è un'operazione semplice: a partire da un elemento radice fittizio, le parole presenti nel file AIML corrispondono ai nodi dell'albero mentre i Template sono gli elementi foglia. Un elemento Pattern è un percorso tra l'elemento radice ed un nodo che precede gli elementi foglia. L'operazione di matching è anch'essa molto intuitiva. Data una frase di input composta da N parole: w_1, w_2, …, w_N la condizione di matching è soddisfatta se sarà presente in un qualsiasi percorso dell'albero la sequenza $w_1, w_2,\ldots w_N$. Questa breve descrizione sulla operazione di matching mostra i limiti del paradigma S/R cioè deve essere presente nell'albero la sequenza w_1, $w_2,\ldots w_N$ senza poter ragionare in termini di similarità semantica tra la parola w_i dell'input e w_n presente nel nodo. Per superare questo limite è stato arricchito ognuno dei nodi del Graphmaster con i synset associati alla parola w_n presente indipendentemente dalla categoria sintattica del termine in esame. Questo semplice approccio rappresenta la BASELINE del lavoro. Per fare ciò è stata implementata una libreria che, sfruttando le interfacce e la libreria JMWNL[6] (Pazienza M.T., A. Stellato, A. Tudorache, 2008), utilizza il lessico strutturato di EuroWordNet (Pazienza M.T., A. Stellato, A. Tudorache, 2008). Ogni termine appartenente al pattern AIML analizzato da ProgramD viene espanso mediante EuroWordNet andando così a formare un insieme di termini sinonimi piuttosto che un termine singolo. In questo modo, durante la fase di matching sarà possibile catturare anche frasi che contengano termini sinonimi a quelli già presenti nel file AIML.
Per valutare tale approccio sono state implementate tre metriche di similarità:

1. **LCH** (Leacock C., M. Chodorow, 1998) trova il più corto path tra due concetti e scala questo valore con la massima lunghezza del path trovata nella gerarchia *is-a* della tassonomia nella quale occorre:

$$sim_{LCH}(c_1, c_2) = -log\frac{len(c_1, c_2)}{2D}(1)$$

dove $len(c_1, c_2)$ è la lunghezza del path più breve tra due concetti e D è la massima profondità della tassonomia.

2. **WUP** (Wu Z. and M. Palmer, 1994) trova la profondità del LCS (Least Common Subsumer) dei concetti e la scala con la somma delle profondità dei concetti individuali (la profondità di un concetto è la sua distanza dal nodo radice):

$$sim_{WUP}(c_1, c_2) = \frac{2N}{N_1 + N_2}(2)$$

dove N, N_1 e N_2 sono le distanze tra nodo root e LCS, concetto c_1 e concetto c_2 rispettivamente.

[6] forniteci dal gruppo ART di Tor Vergata: http://art.uniroma2.it/

3. **PATH** (Pedersen T., S. Patwardhan, J. Michelizzi, 2004) è una metrica di similarità baseline definita come:

$$sim_{WUP}(c_1, c_2) = \frac{1}{path(c_1, c_2)}$$

dove *path* è il percorso più breve tra il concetto c_1 e concetto c_2.

Con queste tre metriche si calcola la similarità tra la parola in input w_i e le parole w_p dei pattern AIML dello stesso livello. Fra tutti i valori che superano un certo valore di threshold, viene scelto quello massimo, che evidenzia quindi un valore maggiore di similarità. Proseguirà, quindi, il matching per il nodo associato al valore scelto. Si è scelto di utilizzare **LCH** come metrica di similarità, fissando il threshold ad 1.8 a fronte dei test eseguiti.

Per utilizzare anche aspetti di semantica distribuzionale si è usata una matrice LSA (Latent Semantic Analysis) formata da 31779 termini per 250 topic, creata utilizzando una finestra di 3 parole per il contesto destro e 3 parole per il contesto sinistro a partire da un corpus di documenti estratti da Wikipedia[7] fornitaci dal gruppo ART [8]. Per calcolare la similarità tra la parola in input w_i e la parola w_p dei pattern AIML si è utilizzata la *cosine similarity*:

$$sim(w_i , w_p) = w_i * w_p / | w_i | * | w_p |$$

tra i vettori della matrice LSA corrispondenti a w_i e w_p.

Anche in questo caso, tra tutti i valori che superano una certa soglia (0.7), si sceglie quello massimo.

Il nuovo spazio vettoriale (le dimensioni sono i topic ed i vettori i termini) sarà utilizzato per valutare la correlazione semantica tra termini con metriche di similarità tra vettori.

4 Testing

Per il testing è stato utilizzato come file di riferimento Religione.aiml[9] in lingua italiana ed appartenente al Package AIML Maria. Questo file consiste di 133 coppie Pattern/Template. Il testing è basato sul confronto tra le risposte della Chatbot con quelle previste da un essere umano e consiste nelle seguenti operazioni:

[7] Il corpus è sviluppato dalla comunità WaCky ed è disponibile nel sito web: http://medialab.di.unipi.it/Project/QA/wikiCoNLL.bz2

[8] http://art.uniroma2.it/ Si ringraziano i dottorandi D. Croce e D. Previtali per aver condiviso i dati (Croce D., D. Previtali, 2010).

[9] http://aitools.org/Free_AIML_sets#Italian_AIML_.28Maria.29

1) costruzione del Testing Set (TS);
2) annotazione manuale del TS (Gold Standard);
3) confronto tra le risposte della Chatbot e quelle del Gold Standard.

Nella fase 1 si sono generate 174 domande in lingua italiana sull'argomento religioso. Per la fase 2 è stato utilizzato un approccio di plausibilità della risposta; pertanto si considerano 3 annotatori che selezionano per ogni domanda, le 3 risposte più plausibili. Si fa presente che questo è un approccio restrittivo che può limitare l'identificazione di risposte esatte tra quelle fornite dal sistema. Per gestire l'agreement tra gli annotatori si è utilizzata la K-Fleiss Statistics (Fleiss I. J., 1971), una metrica utile a stabilire quanto le annotazioni sono tra di loro concordanti e molto flessibile sia alla presenza di annotatori multipli che di multi-classificazione. Tuttavia nel testing, la classificazione è stata binaria (Risposte Rilevanti/Non Rilevanti). Nella fase 3 è eseguito il testing, i cui risultati sono visibili nella *tabella 1*.

Come si può vedere nella prima riga sono presenti i 4 approcci differenti utilizzati. I valori all'interno della tabella rappresentano perciò, per ogni approccio, rispettivamente:

- Numero di domande a cui la chatbot non riesce a rispondere diviso il numero totale di domande sottomesse
- Numero di domande a cui la chatbot risponde in modo corretto diviso il numero totale di domande sottomesse
- Numero di domande a cui la chatbot risponde in modo errato diviso il numero totale di domande sottomesse

	ProgramD (senza estensioni)	BASELINE (espansione EuroWordNet)	BASELINE + METRICHE DI SIMILARITÀ	BASELINE + LSA
Non Riponde	145/174	126/174	126/174	114/174
Risposte Corrette	19/174	29/174	29/174	37/174
Risposte Errate	10/174	19/174	19/174	23/174
TOT.	174/174	174/174	174/174	174/174

Table 1.

5 Analisi dei risultati e conclusioni

L'obiettivo proposto era quello di migliorare la capacità di riconoscere, da parte di una chatbot, un concetto espresso in modo variabile attraverso diverse forme frasali. Dai dati di testing (*tabella 1*) si evince che si è riusciti ad avere un miglioramento di circa il 6% con l'espansione tramite EuroWordNet (BASELINE). In questo caso, la chatbot risponde a 19 domande in più rispetto a ProgramD senza estensioni, di cui 9 errate e 10 corrette. Con la seconda estensione sono state ottenute le stesse prestazioni della precedente a causa del vincolo temporale: la chatbot deve rispondere entro un certo intervallo di tempo, allo scadere del quale la risposta viene considerata errata. L'ultima modalità usata è quella che dà i risultati migliori, infatti viene incrementata la percentuale di risposte corrette dell' 11%. In questo caso, l'incremento di decisioni, rispetto a ProgramD semplice, è stato di 31 risposte, di cui 13 errate e 18 corrette. Nel confronto tra i due approcci (BASELINE e BASELINE+LSA), l'introduzione di LSA permette 12 decisioni in più, con 4 errori commessi. L'incremento di risposte errate è dovuto a due fattori:

- il rischio di errore maggiore nel rispondere a più domande
- la creazione del Test Set mediante una annotazione restrittiva (come già analizzato nella sezione 4 *Testing*)

Tali risultati permettono di pensare con fiducia ad applicazioni industriali.

Inoltre un possibile sviluppo futuro potrebbe essere quello di prevedere un approccio che utilizzi la similarità semantica su base ontologica, in modo da poter sfruttare non solo la distanza tra i concetti, ma anche i valori delle proprietà tra loro in comune.

6 Referenze

Croce D., D. Previtali. (2010). Manifold Learning for the Semi-Supervised Induction of FrameNet Predicates: An Empirical Investigation. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics* (p. 7-16). Sweden: Association for Computational Linguistics.

Fleiss I. J. (1971). Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 378-382.

Leacock C., M. Chodorow. (1998). Combining local context and WordNet similarity for word sense identification. In C. Fellbaum, *WordNet: An electronic lexical database* (p. 265–283). MIT Press.

Pazienza M.T., A. Stellato, A. Tudorache. (2008). A Bottom-up Comparative Study of EuroWordNet and WordNet 3.0, Lexical and Semantic Relations. In K. C. Nicoletta Calzolari (Conference Chair) (A cura di), *LREC.* Marrakesh: European Language Resources Association (ELRA).

Pazienza M.T., A. Stellato, A. Tudorache. (2008). JMWNL: An extensible multilingual library for accessing wordnets in different languages. In K. C. Nicoletta Calzolari (Conference Chair) (A cura di), *LREC.* Marrakesh: European Language Resources Association (ELRA).

Pedersen T., S. Patwardhan, J. Michelizzi. (2004). WordNet::Similarity - Measuring the Relatedness of Concepts. In *Demonstration Papers at HLT-NAACL 2004* (p. 38-41). Boston, Massachusetts: Association for Computational Linguistics.

Wu Z. and M. Palmer. (1994). Verb semantics and lexical selection. In *32nd Annual Meeting of the Association for Computational Linguistics* (p. 133–138). Las Cruces, New Mexico.

DALI Logical Agents into Play

Stefania Costantini, **Annalisa D'Andrea**, **Giovanni De Gasperis**, **Niva Florio**, and **Arianna Tocchio**

Dip. di Ingegneria e Scienze dell'Informazione e Matematica, Università di L'Aquila, Italy

1 Introduction

This paper stems from an experience made by Annalisa D'Andrea (with her colleague Daniela Pavone) as a student of the 'Artificial Intelligence and Intelligent Agents' graduate course at the University of L'Aquila (held by Prof. Stefania Costantini and Dr. Arianna Tocchio). The experience has been further developed by Annalisa as a Ph.D. student, with her collegue Niva Florio and with the help of Dr. Giovanni De Gasperis.

The application that we illustrate here is developed in DALI, an agent-oriented logical language that extends Prolog with reactive and proactive features (see, e.g. [1,2], or [3] for a full list of references on DALI). We show how DALI logical agents and their communication capabilities can be exploited to put into play one's favorite movie or book, where in particular we discuss "Il berretto a sonagli" ("Cap and Bells") by Luigi Pirandello. This by implementing agents that "impersonate" the main characters. In particular, the agents that Daniela and Niva have implemented in DALI do not simply propose chunks of Pirandello's text: rather, each agent is intended to reproduce the main features of the corresponding character's "personality". A particular contribution is given by the DALI communication architecture that provides a meta-level for specifying which messages can be sent or received, from which agents and in which cases. Then, a DALI agent can choose whether to take a message received by some other agent into consideration, or instead discard it. This according to conditions that will be defined depending upon the agent's goals, role, objective and subjective evaluation (the latter according to the agent's "personality"). Symmetrically, the agent is able to decide which agents to "talk" to, and which to exclude from consideration.

These DALI features allowed us to model a basic element of Pirandello's plot: along with the interactions, the characters grow more and more suspicious against each other, and their willingness to communicate decreases until finally arriving to a total blackout. This is rendered by modeling trust management in the communication meta-layer, thus subjecting message exchange to the agent's level of trust in the other agents.

With this example we envisage for future agent-based Artificial Intelligence an extended Turing test, where (in perspective) a human should find it not so easy to understand whether (s)he is talking to a real character or to its agent counterpart. In fact, several students of the Artificial Intelligence and Intelligent Agents graduate course have chosen to model their favorite movie or book in DALI as the subject of their final project, with surprisingly interesting results. In the future we will try to propose "The impossible interviews" ("Le interviste impossibili"), drawing inspiration from a programme dating back to 1975 by RAI2 (one of the three radio networks owned by the Italian government) where a contemporary scholar pretended to interview the "ghost" of some famous person lived in the past. Agents might play the role of the interviewed but possibly also of the interviewer.

The paper is structured as follows. In Section 2 we briefly introduce the DALI language. In Section 3 we summarize some aspects of "Cap and Bells" and how we represent them in DALI. Finally, in Section 4 we conclude.

2 The DALI Language in a Nutshell

DALI [1,2,4] is an Active Logic Programming language designed for executable specification of logical agents. The DALI interpreter is freely available [5]. A DALI agent is a logic program that contains a particular kind of rules, reactive rules, aimed at interacting with an external environment. The environment is perceived in the form of external events, that can be exogenous events, observations, or messages by other agents. In response, a DALI agent can perform actions, send messages, adopt goals, etc. The reactive and proactive behavior of the DALI agent is triggered by several kinds of events: external events, internal, present and past events. It is important to notice that all the events and actions are time-stamped, so as to record when they occurred. The new syntactic items, i.e., predicates related to events and proactivity, are indicated with special postfixes (which are coped with by a pre-processor) so as to be immediately recognized while looking at a program.

2.1 DALI Basic Features

External Events. The external events are syntactically indicated by the postfix *E*. When an event comes into the agent from its "external world", the agent can perceive it and decide to react. Reaction is defined by a reactive rule which has in its head that external event. The special token $:>$, used instead of $:-$, indicates that reactive rules performs forward reasoning. Operationally, incoming events are added to a list called EV and "consumed" according to the arrival order, unless different priorities are specified. Priorities are listed in a separate initialization file including various kinds of directives. By means of these directives the user, in the spirit of [6], can "tune" the agent's behavior under several respects. The advantage of introducing a separate initialization file is that for affecting control there is no need to modify (or even to understand) the agent's main program.

Actions. Actions are the agent's way of affecting her environment, possibly in reaction to an external or internal event. In DALI, actions (indicated with postfix A) may have or not preconditions: in the former case, the actions are defined by actions rules, in the latter case they are just action atoms. An action rule is just a plain rule, but in order to emphasize that it is related to an action, we have introduced the new token $:<$, thus adopting the syntax $action \ :< \ preconditions$. Similarly to events, actions are recorded as past actions.

Internal Events. The internal events define a kind of "individuality" of a DALI agent, making her proactive independently of the environment, of the user and of the other agents, and allowing her to take initiatives, to adopt goals and intentions, to execute plans and to manipulate and revise her knowledge. An internal event is syntactically indicated by the postfix *I*, and its description is composed of two rules. The first one contains the conditions (knowledge, past events, procedures, etc.) that must be true so that the reaction (in the second rule) may happen. Internal events are automatically attempted with a default frequency customizable by means of directives in the initialization file. The user directives can tune several parameters: at which frequency the agent must attempt the internal events; how many times an agent must react to the internal event (forever, once, twice,. . .) and when (forever, when some specific triggering conditions occur, . . .); how long the event must be attempted (until some time, until some terminating conditions, forever). Internal events are treated by the interpreter analogously to external events. A particular kind of internal event is the *goal*, postfix *G*, that stops being attempted as soon as it succeeds.

Present Events. When an agent perceives an event from the "external world", it doesn't necessarily react to it immediately: (s)he can first reason about the event, before (or instead of) triggering a reaction. At this stage, the event is called *present event* and is indicated by the suffix *N*.

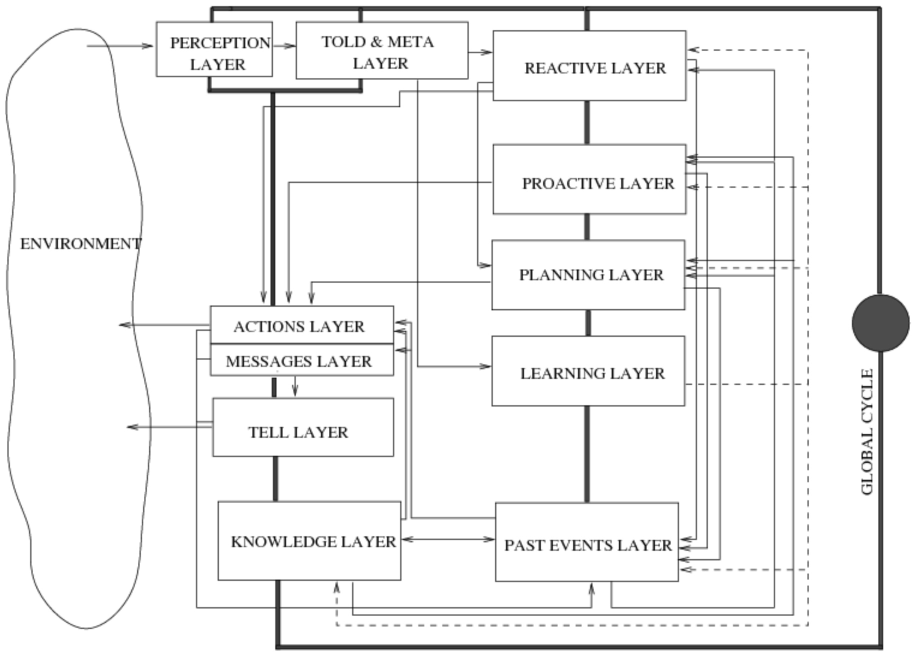


Fig. 1. Architecture of a DALI Agent

Past events Past events represent the agent's "memory", that makes her capable to perform its future activities while having experience of previous events, and of its own previous conclusions. Each past event has a time-stamp T indicating when the recorded event has happened (i.e., when it has been reacted to). Memory of course is not unlimited, neither conceptually nor practically: it is possible to set, for each event, for how long it has to be kept in memory, or until which expiring condition. In the implementation, past events are kept for a certain default amount of time, that can be modified by the user through a suitable directive in the initialization file. Implicitly, if a second version of the same past event arrives, with a more recent time-stamp, the older event is "overridden", unless a directive indicates to keep a number of versions. Old versions of past events are placed in a special structure called PNV, as they may have a role in allowing the agent to reason about the past.

The overall structure of a DALI agent, built out of formerly described elements, is depicted in Figure 1, where the communication components are discussed below.

2.2 DALI Communication Architecture

The DALI communication architecture [7], outlined in Figure 2, consists of four components. The first component (TOLD layer) implements the DALI/FIPA communication protocol and a filter on incoming communication, i.e. a set of rules that decide whether or not to accept reception of a message. The second component is a meta-reasoning user-customizable module that tries to understand message contents, possibly based on ontologies and/or on forms of commonsense reasoning. The third component is the 'core' DALI interpreter. The fourth component implements a filter for the out-going messages. The DALI/FIPA protocol consists of the main FIPA primitives, plus few new primitives which are peculiar of DALI.

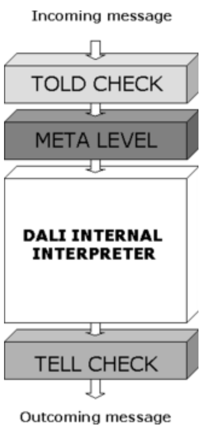


Fig. 2. DALI Communication architecture

When a message is received, it is examined by the TOLD check layer, which is adaptable to the context and modifiable by the user. This filter checks sender and content of the message, and verifies if the conditions for reception are verified. If the conditions are false, this security level eliminates the supposedly wrong message. A similar check is performed to out-going messages via the TELL layer.

DALI communication filters and meta-reasoning on messages are specified by means of meta-level rules defining the distinguished predicates $tell$, $told$ and $meta$ (the latter not discussed here). Some standard rules of general use are pre-defined, to which the user can add new others, specific for the application at hand.

Whenever a message is received, coming from a $Sender$ agent, with content part $Content$, the DALI interpreter automatically looks for a matching $told$ rule. If such a rule is found, the interpreter attempts to prove $told(Sender, Content)$. If this goal succeeds, then the message is accepted, and $Content$ is added to the set of the external events incoming into the receiver agent. Otherwise, the message is discarded. Semantically, this can be understood as implicit reflection up to the filter layer, followed by a reflection down to whatever activity the agent was doing, with or without accepting the message. Symmetrically, messages that an agent sends are subjected to a check via $tell$ rules. For every message that is being sent, the interpreter automatically checks whether an applicable $tell$ rule exists. If so, the message is actually sent only if the goal $tell(Receiver, Content)$ succeeds. Via $tell$ rules one can manage, e.g., mailing lists. In general, via the TELL/TOLD filters useless message exchange can be considerably reduced.

$tell$, $told$ and $meta$ rules are contained in a separate file with respect to the main agent program. Thus, they can be changed without affecting or even knowing the DALI code. The FIPA/DALI communication protocol is implemented by means a piece of DALI code including suitable $tell/told$ rules. This code is in turn specified in a separate file that each DALI agent imports as a library, so that the communication protocol can be seen an “input parameter ”of the agent.

The overall DALI communication architecture is aimed at improving *elaboration tolerance* where, [8]:

> “A formalism is elaboration tolerant to the extent that it is convenient to modify a set of facts expressed in the formalism to take into account new phenomena or changed circumstances. . . .

Communication in DALI is in fact devised in order to be elaboration-tolerant with respect to both the protocol, and the filter.

3 Description of the Application and Demo

As the representation of knowledge and trust plays in general an important role in the interactions among agents, we are going to show in practice how trust can affect the exchange of information among agents. To this purpose, we have defined as a DALI application a small reduction of the Pirandello’s comedy “Il berretto a sonagli” (“Cap and Bells”). Typical for Pirandello is to show how illusion mixes with reality and how people see things in very different way: words are unreliable and everything can be seen at the same time as true and false. In “Il berretto a sonagli” he emphasizes that telling everyone the truth, the sole and cruel truth, regardless of manners or respect and in spite of social habits, will soon produce isolation and a reputation of being mad.

The three main characters of this play have been implemented by three agents of a DALI system: we have not merely reproduced dialogues and lines written by Pirandello, but, based on the dialogues of this play, we have tried to realize agents that mimic the personality of the characters and that reason spontaneously as a spectator would expect.

The title of this play refers to the hat worn by court jesters, that represents the hat of shame to be worn in public. In this comedy, Pirandello pushes the characters into a dilemma with no apparent solution. It is set in Sicily in the early twentieth century and tells the story of Ciampa, a middle-aged bank clerk who lives with his beautiful young wife, Nina, in a small town where sexual infidelity and dishonor are matters of life and death. His boss's wife, Mrs. Beatrice Fiorica, a jealous and dissatisfied woman, becomes convinced (though without proof) that her husband is sleeping with Nina. She is shocked by the discovery of the betrayal and, thinking only of revenge, issues an official "denunciation" of the alleged affair. Although a police investigation turns up no improprieties (also because the police officer Spanò does not want to stand against Beatrice's husband, Cavalier Fiorica), Ciampa's name is still sullied, on the theory that where there is smoke there still must be fire. Ciampa, who paradoxically has always liked to think of himself as the sole voice of reason and sanity in a crazy context, for the sake of respectability does not abandon his wife, but he seriously consider killing her in order to regain his reputation. Luckily for him however, due to Spanò's deception Beatrice is declared legally insane, and is hospitalized for three months in a mental hospital for acting like a fool.

In the practical demonstration of the application, the first act of the comedy is represented. The involved agents communicate through messages. Each agent adopts a communication strategy implying that messages can be exchanged only with trusted agents (in particular, with those for which the level of trust is more than a minimum). At the beginning, each agent trusts the others. Along with the interactions, the level of trust of each agent towards the others will change, and in our particular case it will decrease. In the end, it will become so low as to prevent any further exchange of messages. Trust management for this example is implemented in DALI in a very simple way by means of the following rules:

$$\begin{array}{l} \mathit{told}(\mathit{Sender}, \mathit{Content}) :- \mathit{trust}(\mathit{Sender}, \mathit{Trust}), \mathit{Trust} > k. \\ \mathit{tell}(\mathit{Receiver}, \mathit{Content}) :- \mathit{trust}(\mathit{Receiver}, \mathit{Trust}), \mathit{Trust} > k. \end{array}$$

These rules specify that a certain minimum degree of trust is required in order to enable message exchange. In fact, the first rule accepts reception of incoming messages only if the degree of trust in sender agent is greater than threshold k. The second rule does the same for out-going messages, that are not allowed to be sent to a receiver trusted k or less. Thanks to this mechanism of interaction based on trust, one can leave the agent's "core" program untouched, and still determine an evolution on the behavior of communicating agents.

For the internal reasoning of each agent we widely used internal events. We do not report the full DALI code here, but we just notice that by means of internal events each agent/character "meditates" about her/his circumstances: Beatrice about how to seek revenge, Ciampa about how to restore his good reputation, Spanò about how to avoid to acquire enemies. Based on their internal conclusions, they will then decide to do something. In particular, their internal reasoning process affects the level of trust in other agents. Also, for instance, a character can at a certain point decide that another one is an enemy. This conclusion directly cuts-off communication (even independently of the present level of trust) whenever the following $told$ rule belongs to the TOLD layer:

$$told(Sender, Content) :- not(enemy(Sender)).$$

The demonstration related to this paper shows examples of communication between agents representing Pirandello's characters. It should be possible to notice that the interaction is entirely based on the reconstruction in the DALI system of the Pirandello's character personalities, and that these characteristics determine the generation of events and the corresponding agent answers.

4 Conclusions

The present work can be seen as part of a long-termed experimentation, for which we thank all students of the 'Artificial Intelligence and Intelligent Agents' course: in fact, via the projects that they have to prepare for their exam, they learn how to program agents and at the same time they help us understand how to improve the DALI framework. The specification of the project (cf. the Course web site http://www.di.univaq.it/stefcost/AI2.htm) requires students to choose a movie or a novel of interest for them, and implement as DALI agents the main characters and their interactions. In the opinion of all involved parties, this is an example of how learning can sometimes be fun!

References

1. Costantini, S., Tocchio, A.: A logic programming language for multi-agent systems. In: Logics in Artificial Intelligence, Proc. of the 8th Europ. Conf.,JELIA 2002. LNAI 2424, Springer-Verlag, Berlin (2002)
2. Costantini, S., Tocchio, A.: The DALI logic programming agent-oriented language. In: Logics in Artificial Intelligence, Proc. of the 9th European Conference, Jelia 2004. LNAI 3229, Springer-Verlag, Berlin (2004)
3. Costantini, S.: The DALI agent-oriented logic programming language: References (2012) at URL http://www.di.univaq.it/stefcost/info.htm.
4. Costantini, S., Tocchio, A., Verticchio, A.: Communication and trust in the dali logic programming agent-oriented language. Intelligenza Artificiale, J. of the Italian Association **2**(1) (2005) in English.
5. Costantini, S., D'Alessandro, S., Lanti, D., Tocchio, A.: DALI web site, download of the interpreter (2010) http://www.di.univaq.it/stefcost/Sito-Web-DALI/WEB-DALI/index.php, With the contribution of many undergraduate and graduate students of Computer Science, L'Aquila. For beta-test versions of the interpreter (latest advancements) please ask the authors.
6. Kowalski, R.A.: Algorithm = logic + control. Commun. ACM **22**(7) (1979) 424–436
7. Costantini, S., Tocchio, A., Verticchio, A.: Communication and trust in the DALI logic programming agent-oriented language. Intelligenza Artificiale, J. of the Italian Association of Artificial Intelligence **2**(1) (2005) (in English).
8. McCarthy, J.: Elaboration tolerance. In: Proc. of Common Sense'98. (1998) Available at http://www-formal.stanford.edu/jmc/ elaboration.html.

Artificial Intelligence applied on the Risk! game

Luca Fuligni (luca.fuligni@studio.unibo.it), Andrea Franzon (andrea.franzon@studio.unibo.it), Orfeo Ciano (orfeo.ciano@studio.unibo.it)

Alma Mater Studiorum - Università di Bologna

Abstract. Our goal is to create an application that uses artificial players for the game of Risk!. Some of the existing implementations of Risk! present customizable artificial players but their behavior seems to be neither too smart nor naive and this can tire out the human player. We intent to achieve the goal in a different way: we chose to use Prolog, a declarative language, in contrast with the existing open-source implementations. We learnt that the usage of Prolog reduces the development time and efforts by offering a good abstraction.

Keywords:artificial intelligence, Prolog, Risk, game, dices.

1 Introduction

Our goal is to create an application that uses artificial players for the game of Risk! The context is the usage of the Artificial Intelligences on this game that is a strategy, turn-based, multiplayer game and contains some randomness due to the throwing of dices. More exactly we use A.I. for the game strategy. We choose risk! because unlike games in the theory's games it presents many problematic for example the choice of the attacking territories or the prediction of which territory the enemy will attack.

Usually the existing open-source implementations of this game uses imperative languages to describe the A.I. strategy. We, instead, propose to separate the A.I. aspect from the program by using a declarative logic language. We choose Prolog because it includes the first-order logic and backward chaining mechanism natively. Moreover we don't choose an imperative language, like Java, for the A.I. aspect because all the tree search and inference functionality included in some libraries are less efficient than the Prolog engine.

As a declarative language we choose Prolog instead of other languages like Lisp because we have to infer more on records than on lists. Our intent is to use the Prolog language for describing the artificial player's decisions.

The ability of each artificial player (and so the difficulty for a human player) is characterized by the set of rules that define the inference engine of each artificial player. The particularity of our solution is the connection between prolog that represents the A.I. and Java with which we develop the rest of our application.

2 The game of Risk!

Risk! is a turn-based strategic board game for two to six players in which the goal is indicated by an objective card. The game is played on a board depicting a political map of the Earth that contains 42 territories distributed in 6 continents.

There are several goals but in our application, for simplicity and for a general purpose, we consider only the aim to own 24 territories.

At the start-up each player distributes a fixed number of armies on the board. In each turn there are three phases: placing armies, attacking, fortifying. In the first phase the player puts new armies into the owned territories. In the second the player attack a nearby enemy territory, the success or fail of an attack is decided by the throw of dices. In the third phase the player can move some armies between two owned nearby territories only once. The phases will repeat themselves until a player reaches his goal.

3 Related works

This approach can be used in games which present randomness and multiplayer issues or in games where the artificial player's behavior is strictly connected with logical description like Risk!.

Nowadays there are several implementations of Risk!. The most famous is Risk Digital of Hasbro but there are a lot of Risk! clones like Lux for Linux, Dominion for mobile phones or other flash games can be found over the Internet. Some of those implementations present customizable artificial players but their behavior seems to be neither too smart nor naive and this can tire out the human player.

Our approach, besides focusing on efficiency, challenges the logical aspect in a different and more suitable way in order to improve the game-play.

4 Methodology

Our first implementation is a Java model apt to represent the game board and all the informations that it contains (territory borders, neighborhood relations, continents, territory ownership, ...). We also define the Prolog knowledge-base structure paying attention on data consistency with the Java model.

We propose three different A.I. difficulty levels: easy, medium and hard. The easy level performs random actions. Instead the hard level will be an extension of the medium level that is an "intelligent" implementation of the same Prolog predicates. Each skill that distinguish medium and hard difficulty is mapped into Prolog predicates. We use "GNU Prolog for Java" to embed the Prolog interpreter into our Java application because it respects the ISO standards.

We assume that the aim is to conquer 24 territories to simplify the study of the game. We choose to guide the A.I. through some well-known cases in each phase of the player's turn.

4.1 Knowledge Base

The Risk! Map can be considered as a set of countries (territories). For each territory we define:

- a relation of neighborhood between territories;
- the territory's ownership;
- the number of armies a territory holds;

We represented this knowledge in Prolog by defining the following set of facts:

player/1	represents a player, identified by his color (blue, red, green, pink, black, yellow).
territory/1	represents a territory, identified by his name.
owner/2	represents the ownership of a territory identified by the denoted player.
neighbor/2	represents the relation of the neighborhood between two territories.
army/2	represents the army number for each territory.

So this is the simplest knowledge base we can have:

```
player(red).
player(green).

territory(siberia).
territory(jacuzia).
territory(cita).

owner(siberia,red).
owner(jacuzia, red).
owner(cita,green).

neighbor(siberia,jacuzia).
neighbor(siberia,cita).
neighbor(jacuzia,cita).
neighbor(jacuzia,siberia).
neighbor(cita,jacuzia).
neighbor(cita,siberia).

army(siberia,3).
army(jacuzia,4).
army(cita,2).
```

4.2 Moves

Since a turn is made up of three phases, we define three separated predicates:

place_army/2	Given the player's color, it gives back the territory on which an army should be placed.
attack/3	Given a player's color,it gives back two territories: an attack source (territory owned by the player), and an attack destination (territory owned by an enemy player).
move/3	Given a player's color,it gives back two territories: the territory-source from which the armies should be moved, and the territory-destination to which the armies should be moved.

Here are some sample queries:

```
?- place_army(red,Destination).
Yes, Destination=siberia.
```

```
?- attack(red,Source,Destination).
Yes, Source=yacuzia, Destination=cita.
```

```
?- move(red,Source,Destination).
Yes, Source=siberia,Destination=jacuzia.
```

The implementation of those predicates will vary according to A.I. difficulty. In this way the responsibility of the Java engine is to update the knowledge base and perform the move suggested by the prolog engine.

4.3 Randomness representation in Prolog

Because of randomness component of the game, exploring a decisional tree to choose which territory to attack, could lead to a general low-responsivity. This is because we'd have to insert two randomness level (attack and defense dices) into the decision tree increasing exponentially the number of the nodes in that level.

Therefore we decide to represent the randomness components using a table that contains, given the numer of attacker and defender army, the probability to win an attack. Inside the Knowledge Base in Prolog we represent this information with *victory/3*:

```
victory(Attack#Army, Defense#Army, Probability).
```

So that the predicate "attack",given a fixed threshold value, knew the configuration attacker army/defender army that two territories must have to perform an attack.

4.4 From Java to Prolog

Java has the responsibility of generating the knowledge base. In order to accomplish this task we use the Visitor pattern on the game table (map) object.

```
PrologVisitor visitor = new PrologVisitor();
for(Continent continent:continents)
{
    for(Territory territory:continent.getTerritories())
        visitor.visit(territory);
}
```

Once the knowledge base is updated, it is loaded by the Prolog engine:

```
Environment env = new Environment();
//Loading the knowledge base file and A.I. implementation
env.ensureLoaded(AtomTerm.get("knowledge.pl"));
env.ensureLoaded(AtomTerm.get("easy.pl"));
interpreter = env.createInterpreter();
env.runInitialization(interpreter);
```

The execution of a query:

```
//Calling the query: ?- place_army(red, Territory).
VariableTerm answerTerm = new VariableTerm("Territory");
Term[] args = {AtomTerm.get(playerColor),answerTerm};
CompoundTerm goalTerm = new CompoundTerm(AtomTerm.get("
    place_army"), args);

//Executing the query and testing if succeded
int rc = interpreter.runOnce(goalTerm);
if(rc==PrologCode.SUCCESS || rc==PrologCode.SUCCESS_LAST)
{

    //Getting the Prolog indicated territory from the
        game table
    Term value = answerTerm.dereference();
    String name = TermWriter.toString(value);
    Territory destination = table.getTerritory(name);
}
```

The result of this query will be used by the Risk game engine that performs the corresponding move.

5 Case-study

During an attack, to handle the randomness of the dices, we generate a table (based on Markov chains) which contains the probability of winning an attack by the number of attack and defense armies.

Each player's difficulty is characterized by a threshold value, only if the probability of winning is above this value, the player will attack. Due to this decision sometimes the A.I. reaches a deadlock situation where no one either attacks or moves armies between the territories. To break the deadlock we need to hardcode some specific Prolog conditions.

After thit we again tested the system by letting two players, with the same A.I. difficulty level, fight each-other. Otherwise, if there is at least one player with a different difficulty level, then the specific conditions are not so evident, so the Turing test is passed.

To overcome these problems a finer solution could be to represent the model as a Constraint Linear Programming problem in which every decision is driven by an objective function that will either be minimized or maximized, depending on the action that the player has performed. The objective function varies according to the evaluation of different variables. This evaluation is either rough or refined depending on the player's difficulty level.

6 Conclusions

Our implementation grants a good level of playing for every kind of player. Otherwise the CPL solution would require a fine tuning of the objective function for each difficulty in order to meet the optimal value for each A.I. level. This will produce an unnatural playing style. The Prolog interpreter is an optimal solution for these kinds of problems. By using Prolog it is easier to switch between different A.I. logics (and approaches) leaving the core application unchanged.

References

1. Osborne, Jason A. April 2003: "Markov Chains for the RISK Board Game Revisited", Mathematics Magazine 76
2. S. Russell e P. Norvig, 2005: "Intelligenza artificiale. Un approccio moderno" volume 1 Seconda Edizione.
3. L.Console, E.Lamma, P.Mello, M.Milano, 1997: "Programmazione Logica e Pro-log" Seconda Edizione UTET.

A Neural Network for Automatic Vehicles Guidance.

Alessandro Ghidotti Piovan

Università di Bologna, Italy
alessandro.ghidotti@studio.unibo.it

Abstract. The purpose of this work involves the application and the evaluation of a Reinforcement Learning (RL) based approach to address the problem of controlling the steering of a vehicle. The car is required to drive autonomously on unknown tracks without never going out of the way. The problem has been solved by using a simple neural network with two neurons, and according to the ASE-ACE algorithm. The evaluation has been carried out with referring to The Open Racing Car Simulator (TORCS).

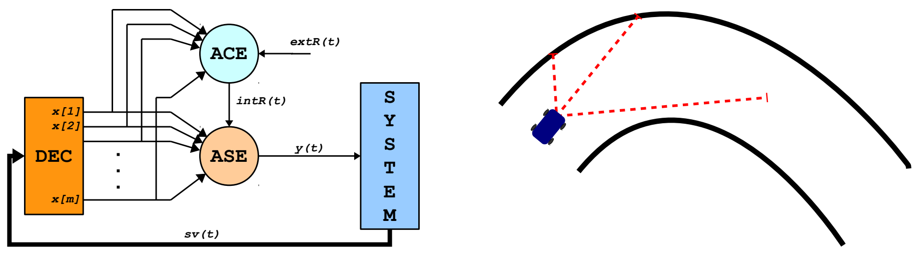
Keywords: reinforcement learning, neural network, car simulator, automatic vehicle guidance

1 Introduction

Reinforcement Learning (RL) is a learning paradigm for neural networks based on the idea of an agent which is placed into a certain environment and it is required it learns to take proper actions without having any knowledge about the environment itself. The agent learns by trial and error: an entity called *critic* observes the agent's behaviour and punishes or rewards the agent if it executes respectively wrong or proper actions; these kinds of judgements on the action taken are said *reinforcements* [1]. Thus, the learning process takes place with the agent beginning to operate into a certain environment with mostly wrong actions, but taking into account reinforcements provided by the critic, and eventually changing its behaviour in order to operate correctly.

A simple neural model for RL implementation is shown in Fig.1(a) and was proposed in [2]. A certain number of state variables $sv(t)$ represent the system state in some time instant. This state variables array is provided as input for a decoder which couples it with the corresponding sensorial stimulus x_i. The decoded stimuli are given as input for (i.e. connected to the synapses of) the two artificial neurons named Associative Search Element (ASE) and Adaptive Critic Element (ACE). Any agent's action depends on the control signal $y(t)$ generated by the ASE. This model distinguishes explicitly *external* from *internal* reinforcement concepts: the former, indicated with $extR(t)$, is the reinforcement signal effectively coming from the critic entity; the latter, indicated with $intR(t)$, is a further and more informative reinforcement coming from the ACE. Notice that this model is based *only* on punishments: the critic never rewards the agent. As the agent's neural network learns the proper behaviour, more and more wrong actions decrease, and the learning curve's convergence slows down; this happens because failures grow away more and more from the taken decision over time. Therefore, the ACE's role is to *predict* whether there will be a failure as a result of an action taken in a certain agent-environment context, not providing to the ASE a "boolean" reinforcement signal, but a finer-graned one which represents the failure probability in that state. The probability is calculated according to the $extR(t)$ value (and some other parameters too [2]), which on the contrary here is a boolean one. The more the state "dangerous" is, the lower the $intR(t)$ value is, and vice versa. Thus, the ASE associates a sensorial stimulus with a control action, and the ACE associates a sensorial stimulus with a failure probability. Finally, ASE-ACE model is really simple to implement and suitable for the automatic vehicles guidance problem too.

Experiments and evaluations have been carried out by using The Open Racing Car Simulator (TORCS) [3]. With respect to this development environment, the system entity shown in Fig.1(a) is the so said *robot*, that is the vehicle chosen for the race.



(a) (b)

Fig. 1. (a) ASE-ACE Neural model. (b) Proximity Sensors model

2 ASE-ACE Applied to the Automatic Vehicles Guidance

2.1 The Problem Model

It's required to provide an intelligent controller for the steering actuator of a vehicle, in such a way that it is able to drive autonomously on any track within a certain category (i.e., road, dirt, snow, etc.). The problem has been modelled in a very simple way which can be schematized as follows:

– The car speed depends on an auxiliary control system, here modeled as a *black-box* on which it is not actually possible to act or get information.

- A set of proximity sensors (i.e. laser range-finders) are installed on the vehicle and each of them provides the distance from the sensor itself and the nearest track boundary into its range [5].
- The steering control output is given by a real value in the range $[-1, 1]$; positive values produce a steering left rotation, a right rotation the negative ones.
- The steering control system does not have any kind of knowledge on the surrounding environment or the physical laws that regulate the car motion.

The aforesaid elements lead to the exclusion of learning paradigms based on training-set or needing a knowledge-base, since the only information available according to the model are the proximity sensors output signals. Therefore, a suitable approach for this problem can be the RL paradigm, just because it does not need other information than the signals coming from the proximity sensors to perform both the training and validation phases.

2.2 Sensors and State Decoding

Since the tracks can have an infinite number of different geometrical characteristics, the choice on the proximity sensors number and their orientation, range, and resolution must be chosen here in an experimental way. The simple and quite good configuration used in this work uses only three sensors as shown in Fig.1(b): the front sensor is oriented as the major axis of the car, while the side ones have an orientation of respectively $\pm 45°$ relative to the major axis of the car. The range of the front sensor is up to 100mt and that of the side sensors is up to 50mt. Notice that range and decoding resolution are key parameters since the algorithm complexity is proportional to the number of possible states (i.e. sensorial stimuli) decoded from the sensors input.

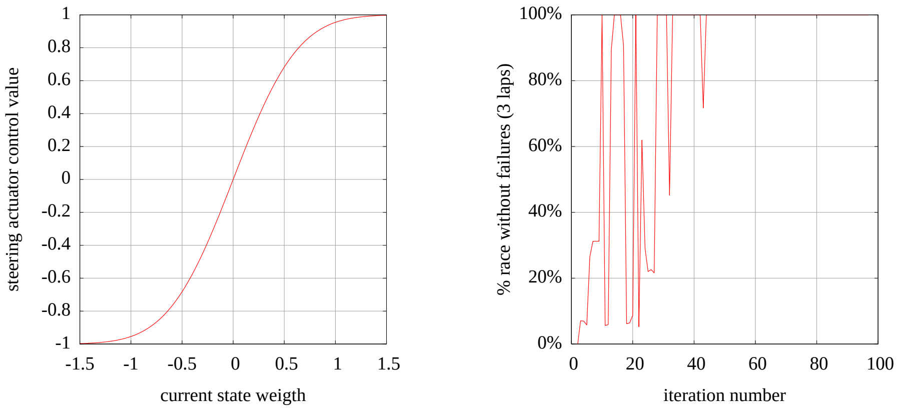
Two non-linear staircase quantization function for the sensors signals were used, each of them with 5 thresholds: the former with ”large” thresholds for the front sensor, the latter with an higher resolution for the side sensors. The rationale is the most the car is in the middle of the track and far away from a turn, the least the state should change; if the car reaches a curve or starts getting close to the track boundary, the state should quickly change to allow to the control system to *quickly* carry out the proper control action.

2.3 Neural Network Output Function

According to the originally proposed model, ASE used the following non linear output function to generate the control action.

$$y(t) = f\left[\sum_{i=1}^{n} w_i(t) + noise(t)\right] \tag{1}$$

where $f(x) = sign(x)$. The aforesaid output function is not suitable to the considered problem because it produces fast commands for the robot steering actuator which correspond to its limit points. This results in a very nervous "driving style", and the car motion assumes a continuous oscillatory trend even on straight sections of the track. Furthermore, because of the acceleration directly handled by the said "black-box", this kind of steering style causes frequent car spinning, especially outgoing from fast curves. For the vehicle guidance problem this output function must be replaced with some kind of "smoother" one. The function chosen here is a sigmoidal function, obtained vertically shifting a zero-mean gaussian cdf with σ^2_{out} variance, as shown in Fig.2(a). This choice has



(a) Output Function. (b) Learning Curve.

Fig. 2. (a) Output function with $\sigma^2_{out} = 0.5$. (b) Learning curve obtained after nearly 100 iterations, each of three laps.

been tested experimentally and it completely eliminated the spinning situations and reduced a lot the vehicle oscillatory trend due to the control noise. Definitely, a set of parameters for the ASE-ACE algorithm which work well in this case are: $\alpha = 10$, $\beta = 0.5$, $\gamma = 0.9$, $\delta = 0.95$, $\lambda = 0.8$, $\sigma_{out} = 0.5$, $\sigma_{noise} = 0.001$; the decoder thresholds are: $4, 6, 8, 12, 16$mt and $4, 6, 7, 10, 12$mt. The aforesaid parameters are defined as follows:

- α and β are positive constants which determine the rate of change of the weights associated with the input pathways for the ASE and the ACE respectively;
- $0 \leq \delta < 1$ and $0 \leq \lambda < 1$ are constants which take part in the computation of the eligibility traces decay rate for the ASE and the ACE respectively;
- $0 < \gamma \leq 1$ is a constant which provides for eventual predictions decay in absence of external reinforcement during the internal reinforcement calculation.

Such parameters have slightly different values if compared with the originally ones used by the authors for the pole-balancing experiment [2], except for the so-said *learning rate* (α). In fact, in this steering control application the number of useful time steps to carry out a trajectory correction should be considerably more than the pole-balancing case, and therefore it's reasonable to choose a lower learning rate.

3 Results and Conclusions

The evaluation of the neural network has been carried out in two phases. Firstly a learning phase in which a totally untrained vehicle starts driving on a set of "easy" learning tracks. Then, a validation phase where the trained vehicle tries driving on a set of unknown tracks. The tracks chosen for the first phase have many different track angles and widths. Cyclically running all the learning tracks, the neural network learns quite well how to control the steering actuator of the vehicle, with respect to the different possible curves properties of a generic track. Each iteration refers to the attempt to perform three complete track's laps. Performing more than one lap leads to speed up the learning curve because the car always begins a race with a standing start from the middle of the track. If the first lap is successfully completed, the car can pass over the start line with very different speed and position. Therefore, after having completed the first lap, the car often goes out of the way at the beginning of the second one. The learning curve obtained during this phase is shown in Fig.2(b), and proofs the learning phase was really fast through this approach, nearly 80 iterations. As shown by the simulation results for the ASE-ACE application to the cart-pole balancing problem [2], after nearly 100 iterations the neural network was able to keep the pole balanced for over 500.000 time-steps (i.e. approximately 2.8 hours of simulated real time), which indicated the learning process completion. Therefore, it is possible to state that the obtained performances are substantially consistent with the authors' experiment ones.

An interesting point is that the learnt driving style favours a central track position on in the width sense. This behaviour is due to the side proximity sensors orientation with respect to the major axis of the car and their range: if the range is smaller, the neural network's state should remain the same within a certain distance from the middle track line. If the side sensors' orientation relative angle is smaller, the vehicle tends to carry out some kind of more sporting trajectories, but with some difficulties on sharp curves; on the contrary, if said angle is greater the vehicle still tends to follow the middle track line, but carrying out forceful steering where it is useless too. A better driving style could be obtained by using seven proximity sensors, where the side ones are oriented with a $30°$ step from the central one [4, 5]. In this case, however, the learning curve convergence speed should be lower because of the much greater number of states of the neural network. After the learning phase, the evaluations on the validation tracks were really good: the vehicle never went out of the way and you could see a slight improvement of lap times too, as long as it converges to a substantially constant time. This improvement of the lap times depends on the track chosen, and although for some tracks there have not been significant improvements, the greatest ones obtained during the experiments have been of nearly 1 second.

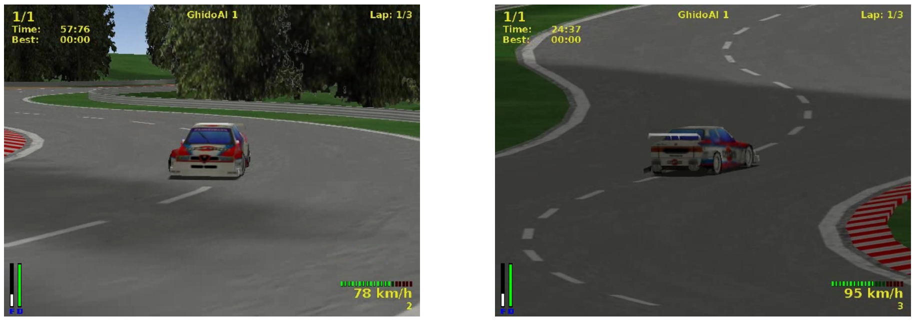


Fig. 3. The robot learnt to drive autonomously on an unknown track with many curves.

References

1. Michie, D., Chambers, RA.: BOXES: An experiment in adaptive control Machine intelligence Vol. 2-2, 137–152 (1968)
2. Barto, A.G., Sutton, R.S., Anderson, C.W.: Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, & Cybernetics, (1983)
3. The open racing car simulator website, `http://torcs.sourceforge.net/`
4. Loiacono, D., Togelius, J., Lanzi, P.L., Kinnaird-Heether, L., Lucas, S.M., Simmerson, M., Perez, D., Reynolds, R.G., Saez, Y.: The wcci 2008 simulated car racing competition. Computational Intelligence and Games, 2008. CIG'08. IEEE Symposium On, 119–126 (2008)
5. Cardamone, L.: On-line and Off-line Learning of Driving Tasks for The Open Racing Car Simulator (TORCS) Using Neuroevolution. Politecnico di Milano, 2008

An Artificial Intelligence that plays for competitive Scrabble

Fulvio Di Maria, Alberto Strade
Alma Mater Studiorum, Bologna
Master Degree in Computer Engineering
fulvio.dimaria@hotmail.it, alberto.strade@live.it

Abstract

An effective program created to play the worldwide famous crossword game SCRABBLE. It uses an efficient data structure that allows fast moves searching possible words trough the lexicon, while an heuristic function determinates the best word to be played, by means of probabilities and sensible considerations. Here is considered the two players version of the game.

1 Introduction

Scrabble is a famous board game, which requires a good knowledge of the native language and a good sense of strategy. On the contrary of other games like Chess and Go, Scrabble is an incomplete information game, since the opponent's rack of tiles is secret.

One of the first computer programs that played Scrabble was MONTY (1), which used both strategic and tactical concepts, but was also a bit slow and never managed to beat Scrabble experts. Others attempt were done, some using a straightforward strategy (playing the longest word or the one which gives more points), others approaching with a bit of protective strategy, like trying not to make available bonus squares to the opponent. In 1988 Appel and Jacobson managed, using a DWAG data structure (Directed Acyclic Word Graph) (2), to create the fastest and most efficient program of their time, surpassed later only by a variant called GALLAD.

While finding all the legal moves is a non-trivial problem on itself, because it implies a CSP (Constraint Satisfaction Problem),(3) being able to choose the best move to perform is equally difficult. Bayes' probability theorem allows to create a good heuristic function (4), choosing the best word evaluating also rack residues.

2 Scrabble Basis

The goal of the game is to make more points than the opponent. The board is composed of a 15X15 square rack where words has to be placed, like in crossword puzzles. There are 120 tiles representing the letters, and each of them reports the value of that letter; there also are 2 blank tiles, which can be used as any letter of the alphabet. Once they are put on the board, they become a copy of the chosen tile and cant be replaced, nor they have a value when calculating the score. Each word must cross at least one letter that was already on the board, and all the perpendicular words that come out from these crosses must be legal words. There are four types of bonus square: 3W triplicates the score of the entire word, 2W doubles it, 3L triplicate the value of the specific letter, while 2L doubles that value. When a player put his word, he draws from the bag until his rack returnes to 7 tiles. In the program at the moment we use a 8 tiles rack, giving human and cpu more possibilities to form longer words and making the game more interesting. Although it implies more computational load, the algorithm is quite efficient, so that this change does not dramatically affect the performance of the program. If a player manages to use all the tiles of the rack, he receives a 50 points bonus. After the tiles of the bag are finished the game turns in a complete information problem, but it ends only when one of the players runs out of tiles. Since we can divide each legal play in across and down, we can talk from now only about the across plays, because the board is symmetric.

3 Structures and Lexicon representation

Any across word must have some newly placed tiles and at least a tile already present on the board. Therefore, we can use this property in order to find legal words to add on the board. Lets call the leftmost newly covered square adjacent to a tile already on the board the *anchorsquare* for that word. So, all the possible anchors are the squares which are adjacent to the filled squares. This way we can reduce the problem of generating all the legal moves by dividing it for each row: for each of it, given the rack, the anchors and the tiles already placed, we can generate all the legal plays.

In order to perform a faster search, we need a structure that stores all the words of the given lexicon (which contains all the words of a dictionary, but also the declinations of the verbs, the plurals and so on). So we used a *trie* (5), a tree which edges are marked by letters. If two words start with the same letters, then they share a portion of their path towards the ending node, which is marked as terminal. Note that all the leaves of the trie are terminal nodes, while the contrary is not always true. This structure can save a lot of words with a memory use proportional to O(n) bytes, resulting in a very efficient solution.

4 Possible words generation

The word generation can be easily divided in two phases: the detection of all the possible tiles anchored to the left of an anchor square (we call these tiles the $leftpart$ of the word); and for each of them, the search of all the possible *right parts* of the words, considering the right part all the tiles at the right of the anchor and the anchor square itself. Consider that the left part can be formed by tiles already on the board, or tiles from the rack, but not both. So, if the square adjacent on the left to the anchor is empty, we must put a left part from the rack (or even have a empty left part), then extend the word starting from the anchor square. If there are any tiles on the left of an anchor we can simply consider that as the left part and perform only the "extend right" phase.

If the left part is all made of tiles already on the board, then we have already the left part, and we have only to note which is; otherwise, we have to find all the left parts. Since the anchor is the leftmost point of adjacency, the left part cant cover an anchor square, and that reduces the length of the left parts of a word anchored to a square, which will be found pruning the trie, according to the constraints of the tiles in our rack.

Once we have all the left parts, we can extend it on the right, adding the tiles one by one, according to the constraints, which are the residual tiles on the rack and the tiles already placed on the board to the right of the anchor square, which must be included in the word (if is not possible, we will immediately delete that istance of right part). Now we have all the possible words that can be played.

5 Heuristic

After the program has determined all the possible words that can be placed, an heuristic function has the duty to choose the most suitable word to play. In the program, the decision is based on 3 parameters, which influence the choice with different weights.

The formula used to express the fitness of each word can be expressed as $F = s - p + c$, where s is the score resulting from playing that word on the board; p is a non negative penalty score, which is greater if the move is weak from a defensive point of view; and c is a combo score, representing the possibility to put a 8 letter word on the table the next turn, which gives a 50 points bonus.

5.1 Penalty Score

Penalty score P is a score which defines how bad is the move from a defensive point of view, in other words it describes the bonus squares that the actual move allows the opponent to use. For each letter it places for the possible word, the program controls along the vertical axis if there are any bonus squares, starting from 7 squares above the word to 7 squares under the word. The malus is higher if it includes a triple score square, decreasing for a double word square, a triple letter square and a double letter square, but it is also conditioned by the distance of that square from the word (since it takes a longer word from the opponent to get advantage of the particular bonus square). Moreover, the malus is then multiplied by a number n, with $0 < n < 1$, which starts from 1 and decreases every time a letter from another word is encountered during the exploration (it takes a lot of effort to get a word long enough to reach the bonus square using a letter already on the board).

The penalty score then is the sum of all the maluses obtained from the exploration of the space up and down of each letter. This number can usually vary from 0 (perfect play from a defensive point of view) to 10 or more. The average value however, is about 6/7.

5.2 Bingo Probability

Before explaining the meaning of the c score, we must define an octet as a string which contains in lexicographic order the letters of a 8-word letter (eg. the word ABBAGLIO has as correspondent optet AABBGILO), and residual rack similarly. The value of c, given that in our lexicon there are over 10000 words with 8 letters, is expressed as

$$c = \sum_{i=0}^{10000} p_i * k$$

where p is the probability of having, after the draw, all the letters of the octet i, and k is a corrective value that is set to 6 after some experimental games. The value of p_i can result easier to understand with an example.

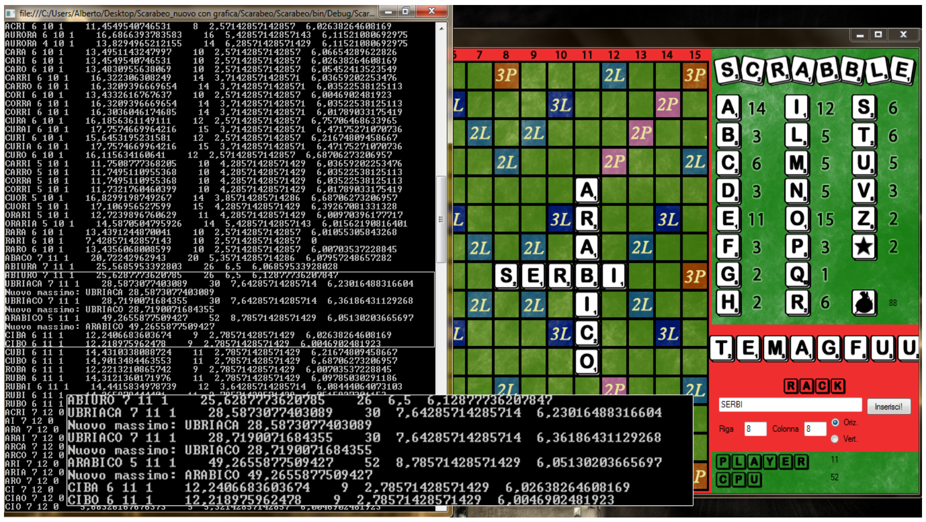
Given a rack, lexicographically ordinate, $\{$ABBEILMU$\}$, assuming that the program is evaluating the fitness of the word $\{$LUME$\}$ as the first word of the game, the residual rack will be $\{$ABBI$\}$. As for the word $\{$ABBAGLIO$\}$, the difference between its octet and the residual rack is $\{$AABBGILO$\}$ $\{$ABBI$\}$ = $\{$AGLO$\}$, so it needs to draw these 4 letters in order to have the complete word in hand. Lets consider now the remaining tiles pile, which is given by all the tiles in the game minus the ones that are on the board and in the residual rack. It will be something like this: $\{$AAAAA...GG..LLLL...OOOOOO...ZZ$\}$. Note that this is not the exact count of the remaining tiles, since we don't nknow which tiles the opponent has in his rack. Given the quantity available of each letter in the tiles pile, the probability of the desired draw is

$$\left[\prod_{i=A}^{Z} \binom{a_i}{d1_i}\right] / \binom{t}{d2}$$

where a is the quantity of tiles of that letter that stills available, $d1$ is the quantity of those tiles that have to be drawn, t is the total number of tiles and $d2$ the number of tiles to draw from the bag. In this case the p_i for the word ABBAGLIO will be $[\binom{9}{1} * \binom{2}{1} * \binom{4}{1} * \binom{12}{1}]/\binom{98}{4}$.

To enhance performances of c, since binomial is not a simple operation, it is possible to create a "binomial matrix" which stores all the binomial coefficients we will need during the game. It is also possible to check before the calculation if there are any chances for that word to be formed drawing from the bag. If there is not, the method skips to the next word and assign p_i=0.

Given this algorithm, is very easy to change it to make it fit with the original version of the Scrabble game, since we have just to change the octet with a septet in the program. The number of 7-letters words is higher than the number of 8-letters ones (15000 against 10000), but the chance to skip a good number of checks makes it possible not to degrade the performances of the program. Actually, the program takes between 3-7



```
ACRI 6 10 1     11,4549540746531     8  2,57142857142857  6,02638264608169
AURORA 6 10 1     16,6866393783583     16  5,42857142857143  6,11521080692975
AURORA 4 10 1     13,8294965212155     14  6,28571428571429  6,11521080692975
CARA 6 10 1     13,4951143247997     10  2,57142857142857  6,06654289622826
CARI 6 10 1     13,4549540746531     10  2,57142857142857  6,02638264608169
CARO 6 10 1     13,4830955638069     10  2,57142857142857  6,05452413523549
CARRI 6 10 1     16,322306308249     14  3,71428571428571  6,03659202253476
CARRO 6 10 1     16,3209396669654     14  3,71428571428571  6,03522538125113
CORI 6 10 1     13,4332616767637     10  2,57142857142857  6,0046902481923
CORRA 6 10 1     16,3209396669654     14  3,71428571428571  6,03522538125113
CORRI 6 10 1     16,3036046174685     14  3,71428571428571  6,01789033175419
CURA 6 10 1     16,1856361149111     12  2,57142857142857  6,75706468633965
CURAI 6 10 1     17,7574669964216     15  3,71428571428571  6,47175271070736
CURI 6 10 1     15,6453195231581     12  2,57142857142857  6,21674809458667
CURIA 6 10 1     17,7574669964216     15  3,71428571428571  6,47175271070736
CURO 6 10 1     16,115634160641     12  2,57142857142857  6,68706273206957
CARRI 5 10 1     11,7508777368205     10  4,28571428571429  6,03659202253476
CARRO 5 10 1     11,7495110955368     10  4,28571428571429  6,03522538125113
CORRA 5 10 1     11,7495110955368     10  4,28571428571429  6,03522538125113
CORRI 5 10 1     11,7321760460399     10  4,28571428571429  6,01789033175419
CUOR 5 10 1     16,8299198749267     14  3,85714285714286  6,68706273206957
CUORI 5 10 1     17,106956527599     15  4,28571428571429  6,39267081331328
ORARI 5 10 1     12,7239896760629     11  4,28571428571429  6,00970396177717
ORARIA 5 10 1     14,5870504795926     14  5,42857142857143  6,01562190816401
RARA 6 10 1     13,4391244870041     10  2,57142857142857  6,01055305843268
RARI 6 10 1     7,42857142857143     10  2,57142857142857  0
RARO 6 10 1     13,4356068008599     10  2,57142857142857  6,00703537228845
ABACO 7 11 1     20,72242962943     20  5,35714285714286  6,07957248657282
ABIURA 7 11 1     25,5685953392803     26  6,5  6,06859533928028
ABIURO 7 11 1     25,6287773620785     26  6,5  6,12877736207847
UBRIACA 7 11 1     28,5873077403089     30  7,64285714285714  6,23016488316604
Nuovo massimo: UBRIACA 28,5873077403089
UBRIACO 7 11 1     28,7190071684355     30  7,64285714285714  6,36186431129268
Nuovo massimo: UBRIACO 28,7190071684355
ARABICO 5 11 1     49,2655877509427     52  8,78571428571429  6,05130203665697
Nuovo massimo: ARABICO 49,2655877509427
CIBA 6 11 1     12,2406683603674     9  2,78571428571429  6,02638264608169
CIBO 6 11 1     12,218975962478     9  2,78571428571429  6,0046902481923
CUBI 6 11 1     14,4310338088724     11  2,78571428571429  6,21674809458667
CUBO 6 11 1     14,9013484463553     11  2,78571428571429  6,68706273206957
ROBA 6 11 1     12,2213210865742     9  2,78571428571429  6,00703537228845
RUBA 6 11 1     14,3121360171976     11  2,78571428571429  6,09785030291186
RUBAI 6 11 1     14,4415834978739     12  3,64285714285714  6,08444064073103
RUBI 6 11 1
RUBO 6 11 1
ACRI 7 12 0
AI 7 12 0
ARA 7 12 0
ARAI 7 12 0
ARCA 7 12 0
ARCO 7 12 0
ARI 7 12 0
ARIA 7 12 0
ARO 7 12 0
CI 7 12 0
CIAO 7 12 0
CIO 7 12 0
```

```
ABIURO 7 11 1     25,6287773620785     26  6,5  6,12877736207847
UBRIACA 7 11 1     28,5873077403089     30  7,64285714285714  6,23016488316604
Nuovo massimo: UBRIACA 28,5873077403089
UBRIACO 7 11 1     28,7190071684355     30  7,64285714285714  6,36186431129268
Nuovo massimo: UBRIACO 28,7190071684355
ARABICO 5 11 1     49,2655877509427     52  8,78571428571429  6,05130203665697
Nuovo massimo: ARABICO 49,2655877509427
CIBA 6 11 1     12,2406683603674     9  2,78571428571429  6,02638264608169
CIBO 6 11 1     12,218975962478     9  2,78571428571429  6,0046902481923
```

Figure 1: Our program in action. Thanks to its heuristic, it has found a word that has a high score (52 points) and does not leave high bonus squares to the opponent.

seconds to choose a move every turn, depending of the openings on the board and the quantity of possible words to test. If the program wouldn't use any heuristic it would take less than a second to find the best word to play (the one that gives the best score).

6 Improvements

Others strategies could be used to enhance quality of play of our program. It could evaluate the usege of large spots of the board, making it difficult for the opponent to find space where to place his word. It could change its strategy according to the score difference between its and the opponent's one. Since the "fitness" score is given as a sum of 3 values, giving each of them a different weight will create different styles of play. Prioritizing the score function, the AI will make more aggressive moves, trying to get as points as possible, leaving, however, good spots for the opponent; giving an higher value to the penalty score we will make the program produce more careful moves, trying not to give the opponent the chance to get a good score; emphasizing the octet function there will be more combo moves, maybe putting shorter words on the table in order to get a "bingo play". With the current set of values, the program can easily reach the 500 points threshold, offering a good challenge even for the most competitive players.

References

[1] Cosma, J.Jasckson: Introducing MONTY plays scrabble. Scrabble Player News, 7-10 (1983)

[2] Appel, Jacobson: The World's Fastest Scrabble Program. Communications of the ACM 31(3), 572-578 (1988)

[3] Russel, Norvig Ariticial Intelligence, a modern approch (3rd Edition). Pearson (2011)

[4] Richards, Amir:Opponent modeling in Scrabble.Proceeding of the Twentieth International Joint Conference on Artificial Intelligence, 1482-1487 (2007)

[5] Fredkin: Trie Memory. CACM 3.9, 490-500 (1960)

Author Index

Baldoni, Matteo, V
Basile, Pierpaolo, 47
Bellini, Anna-Chiara, 60
Bellodi, Elena, 41
Bragaglia, Stefano, 54
Bridi, Thomas, 67
Buffon, Ludovico, 1

Caponi, Emiliano, 72
Cattoni, Roldano, 21
Chesani, Federico, V
Ciano, Orfeo, 86
Corcoglioniti, Francesco, 21
Costantini, Stefania, 79
Cutugno, Francesco, 27

D'Andrea, Annalisa, 79
De Gasperis, Giovanni, 79
de Gemmis, Marco, 47
Derboni, Marco, 7
Di Maria, Fulvio, 98

Fabrini, Irene, 72
Ferraretti, Denis, 14
Florio, Niva, 79
Forment, Davide, 1
Franzon, Andrea, 86
Fuligni, Luca, 86

Gamberoni, Giacomo, 14
Ghidotti Piovan, Alessandro, 92
Girardi, Christian, 21

Lamma, Evelina, 1, 7, 14, 41
Lops, Pasquale, 47

Magnini, Bernardo, V, 21
Mello, Paola, V, 54
Menegatti, Emanuele, 35
Michieletto, Stefano, 35
Montali, Marco, V

Origlia, Antonio, 27

Riguzzi, Fabrizio, 1, 41
Rinaldi, Roberto, 27

Semeraro, Giovanni, 47
Serafini, Luciano, 21
Sottara, Davide, 54
Strade, Alberto, 98

Tocchio, Arianna, 79

Vanzo, Andrea, 72

Zaccaro, Antonio, 7
Zanoli, Roberto, 21
Zese, Riccardo, 41