ASAP: Automatic Semantic Alignment for Phrases

Ana O. Alves
CISUC - University of Coimbra
and Polytechnic Institute of Coimbra
Portugal
ana@dei.uc.pt

Adriana Ferrugento
Mariana Lourenço
Filipe Rodrigues
CISUC - University of Coimbra
Portugal
{aferr,mrlouren}@student.dei.uc.pt
fmpr@dei.uc.pt

Abstract

In this paper we describe the ASAP system (***A**utomatic **S**emantic **A**lignment for **P**hrases*)[1] which participated on the Task 1 at the SemEval-2014 contest (Marelli et al., 2014a). Our assumption is that STS (Semantic Text Similarity) follows a function considering lexical, syntactic, semantic and distributional features. We demonstrate the learning process of this function without any deep preprocessing achieving an acceptable correlation.

1 Introduction

Evaluation of compositional semantic models on full sentences through semantic relatedness and textual entailment, title of this task on SemEval, aims to collect systems and approaches able to predict the difference of meaning between phrases and sentences based on their included words (Baroni and Zamparelli, 2010; Grefenstette and Sadrzadeh, 2011; Mitchell and Lapata, 2010; Socher et al., 2012).

Our contribution is in the use of complementary features in order to learn the function STS, a part of this challenge. Rather than specifying rules, constraints and lexicons manually, we advocate a system for automatically acquiring linguistic knowledge using machine learning (ML) methods. For this we apply some preprocessing techniques over the training set in order to find different types of features. Related to the semantic aspect, we make use of known semantic relatedness and similarity measures on WordNet, in this case, applied to see the relatedness/similarity between phrases from sentences.

Considering the problem of modeling a text corpus to find short descriptions of documents, we aim an efficient processing of large collections while preserving the essential statistical relationships that are useful for, in this case, similarity judgment. Therefore we also apply topic modeling in order to get topic distribution over each sentence set. These features are then used to feed an ensemble algorithm to learn the STS function.

2 Background

2.1 WordNet

WordNet (Miller, 1995) is a computational lexicon of English created and maintained at Princeton University. It encodes concepts in terms of sets of synonyms (called synsets). A synset can be seen as a set of word senses all expressing the same meaning. Each word sense uniquely identifies a single synset. For instance, $car\#n\#1$ uses the notation followed by WordNet and subscript $word\#p\#n$ where p denotes the part-of-speech tag and n the word's sense identifier, respectively. In this case, the corresponding synset $car\#n\#1$, $auto\#n\#1$, $automobile\#n\#1$, $machine\#n\#6$, $motorcar\#n\#1$ is uniquely determined. As words are not always so ambiguous, a word $w\#p$ is said to be *monosemous* when it can convey only one meaning. Alternatively, $w\#p$ is *polysemous* if it can convey more meanings each one represented by a sense number s in $w\#p\#s$. For each synset, WordNet provides the following information: A gloss, that is, a textual definition of the synset; Semantic relations, which connect pairs of synsets. In this context we focus our attention on the Hypernym/Hyponym relation which refers to inheritance between nouns, also known as an *is-a*, or *kind-of* relation and their respective inverses. Y is a hypernym of X if every X is a (kind of) Y (motor_vehicle#n#1 is a hypernym of car#n#1 and, conversely, car#n#1 is

[1] This work was supported by the Crowds project-PTDC/EIA-EIA/115014/2009

a hyponym of vehicle#n#1).

2.2 Semantic similarity

There are mainly two approaches to semantic similarity. First approach is making use of a large corpus and gathering statistical data from this corpus to estimate a score of semantic similarity. Second approach makes use of the relations and the entries of a thesaurus (Lesk, 1986), which is generally a hand-crafted lexical database such as WordNet (Banerjee and Pedersen, 2003). Hybrid approaches combines both methods (Jiang and Conrath, 1997). **Semantic similarity** can be seen as a different measure from **semantic relatedness** since the former compute the proximity between concepts in a given concept hierarchy (e.g. $car\#n\#1$ is similar to $motorcycle\#n$); while the later the common use of both concepts together (e.g. $car\#n\#1$ is related to $tire\#n$).

The Lesk algorithm (Lesk, 1986) uses dictionary definitions (glosses) to disambiguate a polysemous word in a sentence context. The major objective of his idea is to count the number of words that are shared between two glosses, but, sometimes, dictionary glosses are often quite brief, and may not include sufficient vocabulary to identify related sense. In this sense, Banerjee and Pedersen (Banerjee and Pedersen, 2003) adapted this algorithm to use WordNet as the dictionary for the word definitions and extended this metric to use the rich network of relationships between concepts present in WordNet.

The Jiang and Conrath similarity measure (Jiang and Conrath, 1997) computes the information shared between two concepts. The shared information is determined by Information content of the most specific subsume of the two concepts in the hierarchy. Furthermore this measure combines the distance between this subsuming concept and the other two concepts, counting the edge-based distance from them in the WordNet Hypernym/Hyponym hierarchy.

2.3 Topic Modeling

Topic models are based upon the idea that documents are mixtures of topics, where a topic is a probability distribution over words. A topic model is a generative model for documents: it specifies a simple probabilistic procedure by which documents can be generated. To make a new document, one chooses a distribution over topics. Then, for each word in that document, one chooses a topic at random according to this distribution, and draws a word from that topic.

Latent Dirichilet allocation (LDA) is a generative probabilistic topic model of a corpus (Blei et al., 2003). The basic idea is that documents are represented as random mixtures over latent topics, where each topic is characterized by a distribution over words. This process does not make any assumptions about the order of words as they appear in documents. The only information relevant to the model is the number of times words are produced. This is known as the bag-of-words assumption. The main variables of interest in the model are the topic-word distributions Φ and the topic distributions θ for each document.

3 Proposed Approach

Our approach to STS is mainly founded on the idea of learning a regression function that computes that similarity using other variable/features as components. Before obtaining those features, sentences are preprocessed trough known state-of-the-art Natural Language techniques. The resulting preprocessed sentences are then lexically, syntactically and semantically decomposed in order to obtain different partial similarities. These partial similarities are the features used in the supervised learning. These specific stages in our system are explained in detail in the following sections.

3.1 Natural Language Preprocessing

Before computing partial similarities considering different properties of sentences, we need to apply some known Natural Language techniques. For this purpose, we chose OpenNLP[2] as an open-source tool suite which contains a variety of Java-based NLP components. Our focus is here on three core NLP components: tokenization, POS tagging and chunking. Besides the fact OpenNLP also offers a stemmer for English we adopted other implementation self-contained in the specific framework for Topic Modeling (detailed in section 3.3).

OpenNLP is a homogeneous package based on a single machine learning approach, maximum entropy (ME) (Berger et al., 1996). Each OpenNLP tool requires an ME model that contains statistics about the components default features combining diverse contextual information. OpenNLP offers the possibility of both create component or use pre-built models create for different languages.

[2] http://opennlp.sourceforge.net

On one side, components can be trained and customizable models are built for the language and/or domain in study. On the other, the availability of pre-trained models allows the immediate application of such tools on a new problem. We followed the second approach since the sentences are of common-sense and not about a specific domain and are in English[3].

3.2 Feature Engineering

Features, sometimes called attributes, encode information from raw data that allows machine learning algorithms estimate an unknown value. We focus on, what we call, *light* features since they are completely automatic and unsupervised computed, non-requiring a specific labeled dataset for this phase. Each feature is computed as a partial similarity metric, which will later feed the posterior regression analysis. This process is fully automatized, being all features extracted using a pipeline from OpenNLP and other tools that will be introduced in the specific stage where they are used. For convenience and an easier identification in the later machine learning process, we set for each feature an id in the form $f\#n, n \in \{1..65\}$.

3.2.1 Lexical Features

Some basic similarity metrics are used as features related exclusively with word forms. In this set we include: number of negative words[4] for each sentence ($f1$ and $f2$ respectively), and the absolute value of the difference of these counts ($f3 = |f1 - f2|$); the absolute value of the difference of overlapping words for each sentence pair ($f4..7$)[5].

3.2.2 Syntactic Features

OpenNLP tokenization, POS (Part-of-Speech) tagging[6] and text chunking applied on a pipeline fashion allows the identification of (NPs) Noun Phrases, VPs (Verbal Phrases) and (Prepositional Phrases) in sentences. Heuristically, these NPs are further identified as subjects if they are in the beginning of sentences. This kind of shallow parser will be useful to identify the syntactic structure of sentences. Considering only this property, different features were computed as the absolute value of the difference of the number of NPs ($f8$), VPs ($f9$) and PPs($f10$) for each sentence pair.

3.2.3 Semantic Features

WordNet::Similarity (Pedersen et al., 2004) is a freely available software package for measuring the semantic similarity or relatedness between a pair of concepts (or word senses). At this stage we have for each sentence the subject identified as the first NP beginning a sentence.

This NP can be composed of a simple or compound noun, in a root form (lemma) or in a inflected form (plural) (e.g. *electrics* or *economic electric cars*). WorNet::Similarity package also contains a lemmatizer, in the module WordNet::QueryData, which compare a inflected word form and return all WordNet entries which can be the root form of this word. This search is made in all four morphological categories in WordNet (Adjectives, Adverbs, Nouns and Verbs), except when indicated the POS in the end of the queried word, the lemmatizer only see in that specific category (e.g. $flies\#n$ returns $flies\#n, fly\#n$, while $flies$ returns more entries: $flies\#n, fly\#n, fly\#v$). Therefore, a lemmatized is successively applied over the Subjects found for each pair of sentences. The compound subjects are reduced from left to right until a head noun been found as a valid WordNet entry (e.g. the subject $economicelectriccars$ is reduced until the valid entry $electriccar$ which is present on WordNet).

After all the subjects been found and a valid WordNet entry has been matched semantic similarity ($f11$) (Jiang and Conrath, 1997) and semantic relatedness ($f12$) (Lesk, 1986) is computed for each sentence pair. In the case where pair $word\#n$ has multiple senses, the one that maximizes partial similarity is selected.

3.3 Distributional Features

The distribution of topics over documents (in our case, sentences) may contribute to model Distributional Semantic in texts since in the way that the model is defined, there is no notion of mutual exclusivity that restricts words to be part of one topic only. This allows topic models to cap-

[3] OpenNLP offers, for the vast majority of components, at least one pre-trained model for this language.

[4] The Snowball stop word list(Porter, 2001) was used and those words expressing negation were identified (such as: never, not, neither, no, nobody, aren't, isn't, don't, doesn't, hasn't, hadn't, haven't)

[5] Thanks to the SemEval organizers in making available the python script which computes baselines compute_overlap_baseline.py which was applied using different setting for stop word removal, from 0 to 3.

[6] As alternative models are available, the Maxent model with tag dictionary was used on this component. Available at http://opennlp.sourceforge.net/models-1.5/en-pos-maxent.bin

ture polysemy, where the same word has multiple meanings. In this sense we can see topics as natural word sense contexts where words appear in different topics with distinct senses.

Gensim (Řehůřek and Sojka, 2010) is a machine learning framework for Topic Modeling which includes several preprocessing techniques such as stop-word removal and TF-IDF. TF-IDF is a standard statistical method that combines the frequency of a term in a particular document with its inverse document frequency in general use (Salton and Buckley, 1988). This score is high for rare terms that appear frequently in a document and are therefore more likely to be significant. In a pragmatic view, $tf\text{-}idf_{t,d}$ assigns to term t a weight in document d that is: highest when t occurs many times within a small number of documents; lower when the term occurs fewer times in a document, or occurs in many documents; lowest when the term occurs in virtually all documents.

Gensim computes a distribution of 25 topics over sentences not and using TF-IDF ($f13...37$ and $f38...63$). Each feature is the absolute value of the difference of topic$_i$ (i.e. $topic[i] = |topic[i]_{s1} - topic[i]_{s2}|$). Euclidean distance over the difference of topic distribution between sentence pairs in each case (without and with TF-IDF) was also considered as a feature ($f64$ and $f65$).

3.4 Supervised Learning

WEKA(Hall et al., 2009) is a large collection of state-of-the-art machine learning algorithms written in Java. WEKA contains tools for classification, regression, classifier ensemble, and others. Considering the developer version 3.7.11[7] we used the following experiment setup considering the 65 features previously computed for both sentence dataset (train and test) (Marelli et al., 2014b).

One of four approaches is commonly adopted for building classifier ensembles each one focusing a different level of action. Approach A concerns the different ways of combining the results from the classifiers, but there is no evidence that this strategy is better than using different models (Approach B). At feature level (Approach C) different feature subsets can be used for the classifiers, either if they use the same classification model or not. Finally, the data sets can be modified so that each classifier in the ensemble is trained on its own data set (Approach D).

Different methods for generating and combining models exist, like *Stacking* (Seewald, 2002) (Approach B). These combined models share sometimes however the disadvantage of being difficult to analyse, once they can comprise dozens of individual classifiers. Stacking is used to combine different types of classifiers and it demands the use of another learner algorithm to predict which of the models would be the most reliable for each case. This combination is done using a meta-learner, another learner scheme that combines the output of the base learners. The base learners are generally called level-0 models, and the meta-learner is a level-1 model. The predictions of the base learners are input to the meta-learner.

In WEKA, there is a meta classifier called "Stacking".We use this stacking ensemble combining two level-0 models: a K-Nearest Neighbour classifier ($K = 1$) (Aha et al., 1991); and a Linear Regression model without any attribute selection method ($-S1$) and the ridge parameter by default ($1.0 \exp -8$). The meta-classifier was M5P which implements base routines for generating M5 Model trees and rules (Quinlan, 1992; Wang and Witten, 1997).

4 Conclusions and Future Work

Our contribution is in the use of complementary features in order to learn the function of STS, a part of the challenge of building Compositional Distributional Semantic Models. For this we applied some preprocessing tasks over the sentence set in order to find lexical, syntactic, semantic and distributional features. On the semantic aspect, we made use of known semantic relatedness and similarity measures on WordNet, in this case, applied to see the relatedness/similarity between phrases from sentences. We also applied topic modeling in order to get topic distributions over set of sentences. These features were then used to feed an ensemble learning algorithm in order to learn the STS function. This was achieved with a Pearson's r of 0.62780. One direction to follow is to find where the ensemble is failing and try to complement the feature set with more semantic features. Indeed, we plan to explore different topic distribution varying number of topics in order to maximize the log likelihood. Also we would like to select the most relevant feature from this set. We are motivated after this first participation in continuing to improve the system here proposed.

[7] http://www.cs.waikato.ac.nz/ml/weka/downloading.html

References

David W. Aha, Dennis Kibler, and Marc K. Albert. 1991. Instance-based learning algorithms. *Mach. Learn.*, 6(1):37–66.

Satanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, pages 805–810, CA, USA.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP'10)*, pages 1183–1193, PA, USA.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 1394–1404, PA, USA.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. of the Int'l. Conf. on Research in Computational Linguistics*, pages 19–33.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation (SIGDOC '86)*, pages 24–26, NY, USA.

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014a. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. SemEval-2014.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Robertomode Zamparelli. 2014b. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of LREC 2014*.

George A. Miller. 1995. Wordnet: A lexical database for english. *COMMUNICATIONS OF THE ACM*, 38:39–41.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1439.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. Wordnet::similarity: Measuring the relatedness of concepts. In *Demonstration Papers at HLT-NAACL 2004*, HLT-NAACL–Demonstrations '04, pages 38–41, PA, USA.

Martin F. Porter. 2001. Snowball: A language for stemming algorithms. Published online.

Ross J. Quinlan. 1992. Learning with continuous classes. In *5th Australian Joint Conference on Artificial Intelligence*, pages 343–348, Singapore.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the Workshop on New Challenges for NLP Frameworks (LREC 2010)*, pages 45–50, Valletta, Malta.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523.

Alexander K. Seewald. 2002. How to make stacking better and faster while also taking care of an unknown weakness. In C. Sammut and A. Hoffmann, editors, *Nineteenth International Conference on Machine Learning*, pages 554–561.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12)*, pages 1201–1211, PA, USA.

Yong Wang and Ian H. Witten. 1997. Induction of model trees for predicting continuous classes. In *Poster papers of the 9th European Conference on Machine Learning*.