On Why Coarse Class Classification is a Bottleneck for Noun Compound Interpretation

Girishkumar Ponkiya
Dept. of CSE,
IIT Bombay, India
girishp@cse.iitb.ac.in

Pushpak Bhattacharyya
Dept. of CSE,
IIT Bombay, India
pb@cse.iitb.ac.in

Girish K. Palshikar
TCS Research,
Tata Consultancy Services, India
gk.palshikar@tcs.com

Abstract

Sequences of long nouns, i.e., noun compounds, occur frequently and are productive. Their interpretation is important for a variety of tasks located at various layers of NLP. Major reasons behind the poor performance of automatic noun compound interpretation are: (a) lack of a well defined inventory of semantic relations and (b) non-availability of sufficient, annotated, high-quality dataset.

Tratz and Hovy (2010) presented an inventory of semantic relations. They compared existing inventories with their two-level hierarchy, and created a large annotated dataset.

We performed both theoretical as well as data-driven analysis of this inventory. Theoretical analysis reveal ambiguities in the coarse relations. Data-driven analysis report similar performance for coarse as well as fine relations prediction. Our experiments show that improving the coarse classification accuracy can improve the performance of fine class predictor by 13 to 30 points in F-score.

1 Introduction

An important characteristic of a language is the process of creating new words by means of compounding. Especially, in English, technical and scientific literature produces long sequences of nouns, such as *laptop screen, Internet connection, colon cancer symptoms*, etc. Following Downing (1977)'s definition (for English language), these long sequences of nouns, acting as single noun, are known as *noun compounds* (NCs). NLP tasks cannot ignore such long sequences of nouns as they are abundant and productive type of compounds in English.

Most noun compounds appear only once in a large corpus. Baldwin and Tanaka (2004) analyzed the BNC corpus and found that 63.4% of total noun compounds appear only once in the corpus. In addition to the productive nature, noun compounds are compositional. These characteristics of the noun compounds make them a special case, and demand a special treatment.

The conventional approach to tackle this problem is a pipeline of three steps: (1) find noun compounds from text, (2) parse them if required, and (3) extract the semantic relationships between components of the noun compounds. The task of extracting semantic relations between components of a noun compound, or paraphrasing it using verbs and/or prepositions is known as interpretation of noun compound (or noun compound interpretation).

Our primary interest resides in interpretation of two-word noun compounds using predefined semantic labels as classes. The labels have been arranged in a two level hierarchy - coarse classes and fine classes. In this paper, we report the technical and linguistic challenges that we faced while performing classification task. Particularly, we discuss the challenges with coarse level classification.

The rest of the paper is organized as follows: Section 2 covers the related work. Section 3 discusses our approach, the experiments and results for the same are shown in Section 4. Section 5 discusses the results, which is followed by conclusion and future work.

2 Related Work

In computational domain, most work uses either of two representations for semantic relations: (1) set of abstract relations, or (2) paraphrasing a noun compound. Among these two, the former is most popular; repositories of such relations are generally proposed by the linguists.

2.1 Semantic Relations

Researchers have used sets of abstract relations as a preferred choice for semantic relation representation (Levi, 1978; Warren, 1978; Tratz and Hovy, 2010). Levi (1978)'s theory categorizes noun compounds based on the compounding process as: (1) predicate deletion, where a predicate between the components is simply dropped to create a compound, and (2) predicate nominalization, where the head is nominalized form of a verb and modifier is argument of the verb. They proposed a set of abstract predicates for the former category, but no labels for the later category. Later Ó Séaghdha (2007) revised this inventory, and proposed a two level hierarchy of semantic relations.

D S Sharma, R Sangal and A K Singh. Proc. of the 13th Intl. Conference on Natural Language Processing, pages 293–298,
Varanasi, India. December 2016.

	Method	Relations	Examples	Performance
Wermter (1989)	Neural network	7	108	81.5% Accuracy
Lauer (1995)	Various Methods	8	385	47% Accuracy
Rosario et al. (2002)	Rule Based (medical)	18	1660	60% Accuracy
Girju et al. (2005)	WordNet bases rule learning and SVM	35	4205	64% Accuracy
Kim and Baldwin (2005)	WordNet Similarity (k-NN)	20	2169	53% Accuracy
Séaghdha and Copestake (2013)	SVM and String Kernels	6	1443	65% Accuracy
Tratz and Hovy (2010)	MaxEnt / SVM	43	17509	79.4% Accuracy
Dima and Hinrichs (2015)	Deep neural network	43	17509	77.70% F1-score

Table 1: Performance of past approaches for noun compound interpretation. Note that (almost) all methods use different relation inventory and different dataset making it difficult to compare performance.

Levi (1978)'s study is purely based on linguistics. On the other hand, Warren (1978) analyzed the Brown corpus and proposed a four-level hierarchy of semantic relations. Nastase and Szpakowicz (2003) extended Warren (1978)'s approach. Their proposed set of relations is also based on Barker and Szpakowicz (1998)'s semantic relations.

Vanderwende (1994) proposed a set of 13 relations based on the syntactical category and types of questions. Girju et al. (2005) provided another inventory of semantic relation based on Moldovan et al. (2004) for semantic relation in noun phrases.

Finally, Tratz and Hovy (2010) compared and consolidated most of these theories and proposed a two level hierarchy of 43 semantic relations, which are grouped in 9 coarse relations. This inventory of relations was iteratively refined to improve inter-annotator agreement. Tratz and Hovy (2010) used crowd sourcing for the iterative process. We have used this relation repository for our experiments and analysis.

2.2 Automatic Interpretation

Researchers have proposed various methods for automatic interpretation (Wermter, 1989; Nakov, 2013; Dima and Hinrichs, 2015). Unfortunately, these methods have been tested on different relation inventories and datasets, which makes it hard to compare their performance. Table 1 summarizes various methods for automatic interpretation.

All these methods can be categorized in two categories based on how it models the relation: (1) model the relation using only component features (Kim and Baldwin, 2005, 2013), or (2) directly modeling the relation based on how components of a compound can interact in real world (Nakov and Hearst, 2008). A system based on the latter approach should ideally perform better. But, generalization of such information needs large annotated data and a Web scale resource for paraphrase searching.

2.3 Word Embeddings

For classification task, we need to represent a given noun compound as a feature vector. One way is to concatenate the word embeddings of its constituent words. Mathematically, a word embedding is a function $V : D \rightarrow R^n$, where D is a dictionary of the words in a language and R^n is an n-dimensional real space.

Word embeddings are based on the distributional hypothesis. So, the words which occur in similar context have similar vectors. As word vector captures the context, the embedding technique approximates the interaction of a word with other words. This intuition can help us in modeling semantic relation using vectors of the components only.

Dima and Hinrichs (2015) has shown that the noun compound interpretation using only word embeddings, without any feature engineering, gives results comparable to the state-of-the-art. For our experiments, we used Google's pre-trained word embeddings[1] (Mikolov et al., 2013a,b).

3 Approach

Our aim is to classify a given noun compound into one of fine classes. The classes are arranged in two level hierarchy. We want to exploit the hierarchy to improve the system.

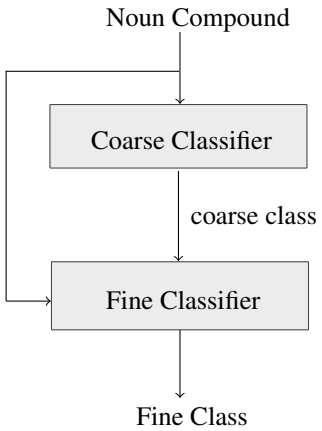

Figure 1: Pipeline architecture

If the coarse class of a noun compound is known, then inter coarse class confusion can be avoided. We try to leverage this fact by proposing a pipeline architecture (refer Figure 1) for the classification task. We

[1] https://code.google.com/archive/p/word2vec/

System Type	Input	Output	Remark
Type-1	vectors of the components + 1-hot presentation of a coarse relations	43 fine relations	
Type-2	vectors of the components + coarse class number	43 fine relations	
Type-3	vectors of the components only	9 coarse relations	Coarse Classification
Type-4	vectors of two components only	43 fine relations	End-to-end

Table 2: System types based on input features and output classes/relations.

define various systems based on the input and output to experiment with.

4 Experiments and Results

In this section we explain dataset creation, experiment setup, and the results. An analysis of experiments is discussed in the next section.

4.1 Dataset Pre-processing

For our experiment, we used Tratz and Hovy (2010)'s relation inventory and dataset. This inventory has 43 fine relations, grouped under 9 coarse relations. In this dataset, each example has been labeled with one of 43 fine semantic relations. We can get coarse class label indirectly as each fine relation belongs to exactly one coarse relation.

There are totally 19036 examples of noun-noun compounds in the Tratz and Hovy (2010)'s dataset. Some examples in the dataset contain more than one word as a component, e.g., *health-care legislation, real-estate speculation*. We eliminated such examples. We also eliminate examples for which at least one components has no word vector. This result in 18857 examples. We used Google's pre-trained word vectors to create feature vectors. For example, feature vector of *passenger traffic* is concatenation of two vectors: vector of *passenger* and vector of *traffic*.

We shuffled our dataset, and split it into three disjoint sets: train set (65%), validation set (15%), and test set (20%). The system was trained on the train set. The validation set was used for validation and hyper-parameter searching. The system was evaluated on the test set.

4.2 Experimental Setup

To check our hypothesis, we defined four types of systems. Table 2 defines system types based on the input feature vector and the output classes. Given the word embeddings of a noun compound, Type-3 and Type-4 systems predict a coarse relation and a fine relation, respectively.

Type-1 and Type-2 systems are modeled as follows: given a noun compound and true coarse relation, predict a fine relation. The sole difference between them is how coarse-class information is represented. It is represented as 1-hot encoding in Type-1 system, whereas as a single numeric value in Type-2.

We used the following classifiers for our experiment with the above mentioned four types of settings:

- Naive Bayes
- k-nearest neighbor (kNN)
- SGD: Various linear classifiers with standard loss functions and stochastic gradient descent optimization (implemented as SGD in scikit-learn (Pedregosa et al., 2011))
- Decision tree (ID3 algorithm)
- Support vector machine (SVM)
- Deep neural network (DNN) (similar to Dima and Hinrichs (2015)'s architecture)

In addition to the above four types of systems and the classifiers, we use a pipeline architecture (proposed in Section 3), where predicted output of Type-3 system (coarse relation predictor) was fed to the Type-1 or Type-2 system.

For all of these multi-class classifications, we used one-vs-one and one-vs-rest techniques. Performance of the system in both sets of techniques are very similar.

4.3 Results

Figure 2 summarizes the results of various classifiers with different settings. SVM with polynomial kernel (degree=2, and soft margin constant C=10) outperforms all classifiers for all the mentioned systems. SVM results are separately shown in Table 3.

System Type	Precision	Recall	F1-score
Type-1	0.93	0.93	0.93
Type-2	0.92	0.92	0.92
Type-3	0.84	0.85	0.84
Type-4	0.81	0.81	0.80
Pipeline	0.79	0.8	0.79

Table 3: SVM (polynomial, degree 2, C 10) Results for all system types.

There are some interesting patterns across all classifiers. Most important patterns are: (a) performance of Type-3 (coarse predictor) and Type-4 (fine predictor) are almost same for most classifiers, (b) when coarse relation information was fed to a classifier in addition to word vectors (Type-1 or Type-2), performance of the system boosts up by 13 to 30 points in F-score.

For pipeline architecture, overall system performance degrades slightly compared to Type-4 system.

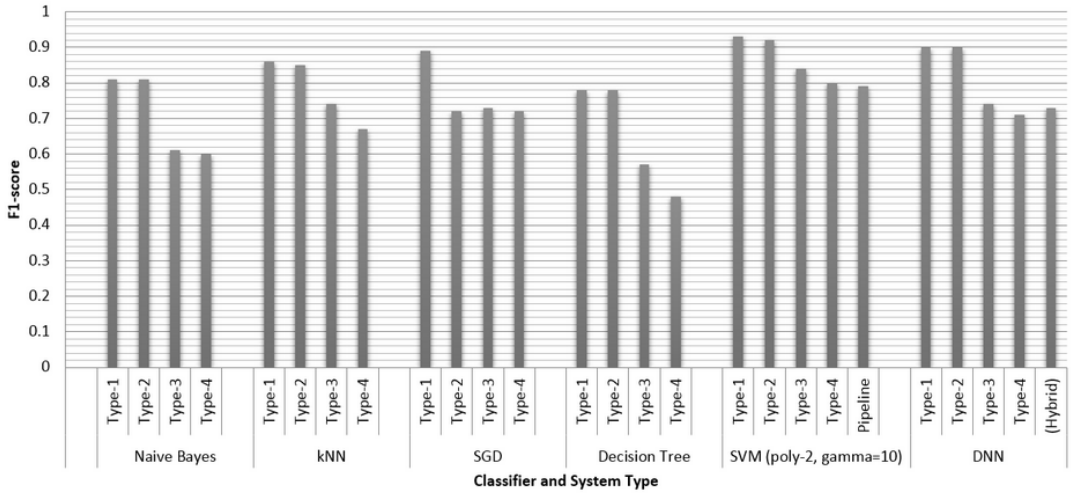

Figure 2: Performance (**F-score**) of combination of classifiers and system types. The system types are defined in Table 2.

Results for pipeline architecture (with SVM classifier) are shown in Figure 2. Similarly, slightly less F-scores are observed for other classifiers.

5 Discussions

Typically, it is expected that prediction among fine classes is difficult compared to prediction among coarse classes. A fine class predictor requires more information/samples compared to coarse class predictor. Particularly, the number of required training examples increases as the number of parameter to be estimated increases. We have the same number of examples for coarse and fine prediction. So, coarse classification is supposed to outperform fine classification. Despite such expectation, there is no significant difference in Type-3 (coarse predictor) and Type-4 (fine predictor) with most classifiers.

To check whether a better coarse class predictor can help the system in predicting the fine class, we used true coarse class labels as features in the fine class predictor (Type-1 and Type-2). Additional coarse class information boosts the performance of fine class predictor by 13 to 30 points in F1-score. This is evident from the difference in performance of Type-1 and Type-2 systems as compared to Type-4 system shown in Figure 2.

We analyzed the semantic relations and annotated dataset to understand the main reasons behind the poor performance of coarse relation predictor. Following are some important observations:

- The definition of OTHER is too vague. There are two fine relations under this coarse relation: *Other* and *Fixed Pair / Special / Lexicalized*. For '*Other*' subcategory, the annotator guideline says "*relationship between the two words is fairly clear, but no category seems appropriate*."

	Precision	Recall	F1-score	# Test Sample
C1	0.79	0.78	0.79	217
C2	0.84	0.91	0.88	1275
C3	0.78	0.78	0.78	361
C4	0.89	0.84	0.86	95
C5	0.79	0.72	0.76	258
C6	0.79	0.77	0.78	232
C7	0.89	0.90	0.90	867
C8	0.90	0.84	0.87	393
C9	0.55	0.14	0.23	77

Table 4: Confusion matrix for the coarse class predictor (Type-3) using SVM. (**C1**: "Causal Group", **C2**: "Function / Purpose Clusters", **C3**: "Ownership, Employment, and Use", **C4**: "Time", **C5**: "Location & Whole+Part", **C6**: "Composition / Containment", **C7**: "Topical", **C8**: "Coreferential / Attributive", **C9**: "Other")

	C1	C2	C3	C4	C5	C6	C7	C8	C9
C1	170	20	11	0	5	2	9	0	0
C2	15	1166	19	3	13	12	32	7	8
C3	10	38	281	1	8	6	11	6	0
C4	1	7	1	80	0	0	2	4	0
C5	4	18	18	0	187	9	19	3	0
C6	2	31	5	0	5	179	4	6	0
C7	6	43	13	5	8	4	781	6	1
C8	4	21	6	0	8	12	12	330	0
C9	3	38	4	1	3	3	8	6	11

Table 5: Confusion matrix for the coarse class predictor (Type-3) using SVM. For the labels, refer Table 4.

- PURPOSE/ACTIVITY GROUP has overlap with most of the coarse classes. This is also seen in the confusion matrix (ref. 2nd column and 2nd row in Table 5). In proportion to the total examples in this class, column values are small. But, as this class covers 33% of the total examples, a "small" value in C2 column can penalize the recall of other classes heavily (like C9). The same

holds true for the row.

- *Fixed Pair / Special / Lexicalized* are multiword expressions, and such expressions should be handled separately. Typically, we can interpret compositional compounds only. If we feed a multiword expression to a system which is trained to interpret noun-noun compounds, then the system will try to predict the semantic relation based on semantics of the components. This was clear from error analysis and the confusion matrix. The last row (ref. Table 5) is heavy as compared to last column.

- *Partial Attribute/Feature/Quality Transfer*, belonging to ATTRIBUTIVE AND COREFERENTIAL coarse relation, is defined as "*The {attribute2} is metaphorically (but not literally) a/the/made_of {attribute1}*." Like multiwords, the interpretation of metaphoric noun compounds cannot be inferred from the composition of the semantics of its components.

6 Conclusion and Future Work

In this paper, we claimed that the main bottleneck in noun compound interpretation task is the ambiguity between coarse classes. We analyzed an inventory of semantic relations and annotated data to understand such ambiguities. In addition, we also used a data driven approach to show that coarse relation prediction is a major bottleneck in the system. Experiment results also show that if one can improve coarse predictor, then overall system will improve significantly.

The SVM classifier with polynominal kernel (degree=2, and C=10) outperforms Tratz and Hovy (2010) and Dima and Hinrichs (2015) by a small margin. The salient points of our system are: 1) our system uses only word vectors, and 2) our system does not rely on computationally complex algorithms and resources.

To solve the ambiguity in semantic relation, we observe two possible directions:

1. Separate interpretable semantic relations from the rest, and define (or reshape) the coarse relation boundary accordingly.

2. Refining the definitions by adding more information. Such revised definitions may help the researcher in clarifying the class boundary. Such information can later be used to make the system more robust.

Acknowledgments

We thank CFILT members at IIT Bombay for their valuable comments and suggestions. This work is funded by Tata Consultancy Services Limited (TCS) under NGIE (Next Generation Information Extraction) with project code 13TCSIRC001.

References

Baldwin, T. and Tanaka, T. (2004). Translation by machine of complex nominals: Getting it right. In *Proceedings of the Workshop on Multiword Expressions: Integrating Processing*, pages 24–31. Association for Computational Linguistics.

Barker, K. and Szpakowicz, S. (1998). Semi-automatic recognition of noun modifier relationships. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 96–102. Association for Computational Linguistics.

Dima, C. and Hinrichs, E. (2015). Automatic noun compound interpretation using deep neural networks and word embeddings. *IWCS 2015*, page 173.

Girju, R., Moldovan, D., Tatu, M., and Antohe, D. (2005). On the semantics of noun compounds. *Computer speech & language*, 19(4):479–496.

Kim, S. N. and Baldwin, T. (2005). Automatic interpretation of noun compounds using wordnet similarity. In *Natural Language Processing–IJCNLP 2005*, pages 945–956. Springer.

Kim, S. N. and Baldwin, T. (2013). A lexical semantic approach to interpreting and bracketing english noun compounds. *Natural Language Engineering*, 19(03):385–407.

Lauer, M. (1995). Corpus statistics meet the noun compound: some empirical results. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 47–54. Association for Computational Linguistics.

Levi, J. N. (1978). *The syntax and semantics of complex nominals*. Academic Press New York.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751.

Moldovan, D., Badulescu, A., Tatu, M., Antohe, D., and Girju, R. (2004). Models for the semantic classification of noun phrases. In *Proceedings of the HLT-NAACL Workshop on Computational Lexical Semantics*, pages 60–67. Association for Computational Linguistics.

Nakov, P. (2013). On the interpretation of noun compounds: Syntax, semantics, and entailment. *Natural Language Engineering*, 19(03):291–330.

Nakov, P. and Hearst, M. A. (2008). Solving relational similarity problems using the web as a corpus. In *ACL*, pages 452–460. Citeseer.

Nastase, V. and Szpakowicz, S. (2003). Exploring noun-modifier semantic relations. In *Fifth international workshop on computational semantics (IWCS-5)*, pages 285–301.

Ó Séaghdha, D. (2007). Annotating and learning compound noun semantics. In *Proceedings of the 45th Annual Meeting of the ACL: Student Research Workshop*, pages 73–78. Association for Computational Linguistics.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Rosario, B., Hearst, M. A., and Fillmore, C. (2002). The descent of hierarchy, and selection in relational semantics. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 247–254. Association for Computational Linguistics.

Séaghdha, D. O. and Copestake, A. (2013). Interpreting compound nouns with kernel methods. *Natural Language Engineering*, 19(03):331–356.

Tratz, S. and Hovy, E. (2010). A taxonomy, dataset, and classifier for automatic noun compound interpretation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 678–687. Association for Computational Linguistics.

Vanderwende, L. (1994). Algorithm for automatic interpretation of noun sequences. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pages 782–788. Association for Computational Linguistics.

Warren, B. (1978). Semantic patterns of noun-noun compounds. *Acta Universitatis Gothoburgensis. Gothenburg Studies in English Goteborg*, 41:1–266.

Wermter, S. (1989). Learning semantic relationships in compound nouns with connectionist networks. In *Program of the Eleventh Annual Conference of the Cognitive Science Society*, pages 964–971.