Luminoso at SemEval-2018 Task 10: Distinguishing Attributes Using Text Corpora and Relational Knowledge

Robert Speer
Luminoso Technologies, Inc.
675 Massachusetts Avenue
Cambridge, MA 02139
rspeer@luminoso.com

Joanna Lowry-Duda
Luminoso Technologies, Inc.
675 Massachusetts Avenue
Cambridge, MA 02139
jlowry-duda@luminoso.com

Abstract

Luminoso participated in the SemEval 2018 task on "Capturing Discriminative Attributes" with a system based on ConceptNet, an open knowledge graph focused on general knowledge. In this paper, we describe how we trained a linear classifier on a small number of semantically-informed features to achieve an F_1 score of 0.7368 on the task, close to the task's high score of 0.75.

1 Introduction

Word embeddings are most effective when they learn from both unstructured text and a graph of general knowledge (Speer and Lowry-Duda, 2017). ConceptNet 5 (Speer et al., 2017) is an open-data knowledge graph that is well suited for this purpose. It is accompanied by a pre-built word embedding model known as ConceptNet Numberbatch[1], which combines skip-gram embeddings learned from unstructured text with the relational knowledge in ConceptNet.

A straightforward application of the ConceptNet Numberbatch embeddings took first place in SemEval 2017 task 2, on semantic word similarity. For SemEval 2018, we built a system with these embeddings as a major component for a slightly more complex task.

The Capturing Discriminative Attributes task (Paperno et al., 2018) emphasizes the ability of a semantic model to recognize relevant differences between terms, not just their similarities. As the task description states, "If you can tell that americano is similar to capuccino and espresso but you can't tell the difference between them, you don't know what americano is."

The ConceptNet Numberbatch embeddings only measure the similarity of terms, and we hypothesized that we would need to represent more specific relationships. For example, the input triple "frog, snail, legs" asks us to determine whether "legs" is an attribute that distinguishes "frog" from "snail". The answer is yes, because a frog *has* legs while a snail does not. The *has* relationship is one example of a specific relationship that is represented in ConceptNet.

To capture this kind of specific relationship, we built a model that infers relations between ConceptNet nodes, trained on the existing edges in ConceptNet and random negative examples. There are many models designed for this purpose; the one we decided on is based on Semantic Matching Energy (SME) (Bordes et al., 2014).

Our features consisted of direct similarity over ConceptNet Numberbatch embeddings, the relationships inferred over ConceptNet by SME, features that compose ConceptNet with other resources (WordNet and Wikipedia), and a purely corpus-based feature that looks up two-word phrases in the Google Books dataset.

We combined these features based on ConceptNet with features extracted from a few other resources in a LinearSVC classifier, using liblinear (Fan et al., 2008) via scikit-learn (Pedregosa et al., 2011). The classifier used only 15 features, of which 12 ended up with non-zero weights, from the five sources described. We aimed to avoid complexity in the classifier in order to prevent overfitting to the validation set; the power of the classifier should be in its features.

The classifier produced by this design (submitted late to the contest leaderboard) successfully avoided overfitting. It performed better on the test set than on the validation set, with a test F_1 score of 0.7368, whose margin of error overlaps with the evaluation's reported high score of 0.75.

At evaluation time, we accidentally submitted our results on the validation data, instead of the

[1] https://github.com/commonsense/conceptnet-numberbatch

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 985–989
New Orleans, Louisiana, June 5–6, 2018.

test data, to the SemEval leaderboard. Our code had truncated the results to the length of the test data, causing us to not notice the mismatch. This erroneous submission got a very low score, of course. This paper presents the corrected test results, which we submitted to the post-evaluation CodaLab leaderboard immediately after the results appeared. We did not change the classifier or data; the change was a one-line change to our code for outputting the classifier's predictions on the test set instead on the validation set.

2 Features

In detail, these are the five sources of features we used:

ConceptNet vector similarity. Given the triple ($term_1$, $term_2$, att), we look up the ConceptNet Numberbatch embeddings for the root words of the three terms (with root words determined using ConceptNet's built-in lemmatizer). We determine the cosine similarity of ($term_1$, att) and the cosine similarity of ($term_2$, att). We then subtract the square roots of the similarity scores (floored at 0). If this difference is large enough, it indicates a positive example, a discriminative attribute that applies to $term_1$ and not to $term_2$.

ConceptNet relational inference. We train a Semantic Matching Energy model to represent ConceptNet nodes and relations as vectors, along with a 3-tensor of interactions between them. This model can then assign a confidence score to any triple (a relation connecting two terms). We used this model to infer values for each of 11 different ConceptNet relations. As in the case of vector similarity, each feature value is the difference between the value inferred for *rel*($term_1$, att) and *rel*($term_2$, att). This model is described in more detail in the next section.

Wikipedia lead sections. This feature expands on ConceptNet vector similarity: instead of computing the similarity between the attribute and the term, it computes the maximum of the similarity between the attribute and any word that appears in the lead section of the Wikipedia article for the term (Wikipedia, 2017). This helps to identify attributes that would be used to define the term, such as "amphibian" as an attribute for "frog".

WordNet entries. This feature is similar to the "Wikipedia lead sections" feature. It expands each term by looking up its synonyms in WordNet (Miller et al., 1998), the synonyms in synsets it is connected to, and the words in its gloss (definition), and taking the maximum similarity of the attribute to any of these terms.

Google Books 2-grams. This feature determines if $term_1$ forms a significant two-word phrase with att, more than $term_2$ does, based on the Google Books English Fiction data (Lin et al., 2012). The "significance" (s) of a two-word phrase is determined by comparing the smoothed log-likelihood of the individual unigrams to the smoothed log-likelihood of the phrase:

$$\begin{aligned}\mathrm{s}(term, att) = 10 + \log_{10}(\#(term, att) + 1) \\ - \log_{10}((\#(term) + 10^5)(\#(att) + 10^5))\end{aligned}$$

where $\#$ represents the number of occurrences of a unigram or bigram in the corpus.

The "ConceptNet relational inference" feature provides 11 entries to the feature vectors, while the other sources each provide one. In total, there are 15 features that represent each input triple.

Across multiple data sources, we use the square root of cosine similarity to measure the strength of the match between a term and an attribute. Because attributes should be at least somewhat related to the terms they describe, and because weak semantic similarity can be interpreted as relatedness, the square root helps us emphasize the important part of the scale. The difference between "somewhat related" and "not related" is more important to the task than the difference between "very similar" and "somewhat related", as a discriminative attribute should ideally be unrelated to the second term.

2.1 The Relational Inference Model

To infer truth values for ConceptNet relations, we use a variant of the Semantic Matching Energy model (Bordes et al., 2014), adapted to work well on ConceptNet's vocabulary of relations. Instead of embedding relations in the same space as the terms, this model assigns new 10-dimensional embeddings to ConceptNet relations, yielding a compact model for ConceptNet's relatively small set of relations.

The model is trained to distinguish positive examples of ConceptNet edges from negative ones. The positive examples are edges directly contained in ConceptNet, or those that are entailed by changing the relation to a more general one or

switching the directionality of a symmetric relation. The negative examples come from replacing one of the terms with a random other term, the relation with a random unentailed relation, or switching the directionality of an asymmetric relation.

We trained this model for approximately 3 million iterations (about 4 days of computation on an nVidia Titan Xp) using PyTorch (Paszke et al., 2017). The code of the model is available at `https://github.com/LuminosoInsight/conceptnet-sme`.

To extract features for the discriminative attribute task, we focus on a subset of ConceptNet relations that would plausibly be used as attributes: *RelatedTo*, *IsA*, *HasA*, *PartOf*, *CapableOf*, *UsedFor*, *HasContext*, *HasProperty*, and *AtLocation*.

For most of these relations, the first argument is the term, and the second argument is the attribute. We use two additional features for *PartOf* and *AtLocation* with their arguments swapped, so that the attribute is the first argument. The generic relation *RelatedTo*, unlike the others, is intended to be symmetric, so we add its value to the value of its swapped version and use it as a single feature.

3 The Overfitting-Resistant Classifier

The classifier that we use to make a decision based on these features is scikit-learn's LinearSVC, using the default parameters in scikit-learn 0.19.1. (In Section 4, we discuss other models and parameters that we tried.) This classifier makes effective use of the features while being simple enough to avoid some amount of overfitting.

One aspect of the classifier that made a noticeable difference was the scaling of the features. We tried L_1 and L_2-normalizing the columns of the input matrix, representing the values of each feature, and decided on L_2 normalization.

We took advantage of the design of our features and the asymmetry of the task as a way to further mitigate overfitting. All of the features were designed to identify a property that $term_1$ has and $term_2$ does not, as is the case for the discriminative examples, so they should all make a non-negative contribution to a feature being discriminative. We can inspect the coefficients of the features in the SVC's decision boundary. If any feature gets a negative weight, it is likely a spurious result from overfitting to the training data. So, after training the classifier, we clip the coefficients of the decision boundary, setting all negative coefficients to zero.

Feature	Coefficient
ConceptNet vector similarity	13.82
SME: RelatedTo	14.01
SME: (x IsA a)	2.13
SME: (x HasA a)	0.00
SME: (x PartOf a)	0.56
SME: (x CapableOf a)	3.72
SME: (x UsedFor a)	0.92
SME: (x HasContext a)	0.88
SME: (x HasProperty a)	0.00
SME: (x AtLocation a)	0.00
SME: (a PartOf x)	3.22
SME: (a AtLocation x)	0.69
Wikipedia lead sections	12.46
WordNet relatedness	13.95
Google Ngrams	28.82

Table 1: Coefficients of each feature in our linear classifier. x represents a term and a represents the attribute.

If we were to remove these features and re-train, or require non-negative coefficients as a constraint on the classifier, then other features would inherently become responsible for overfitting. By neutralizing the features *after* training, we keep the features that are working well as they are, and remove a part of the model that appears to purely represent overfitting. Indeed, clipping the negative coefficients in this way increased our performance on the validation set.

Table 1 shows the coeffcients assigned to each feature based on the training data.

4 Other experiments

There are other features that we tried and later discarded. We experimented with a feature similar to the Google Books 2-grams feature, based on the AOL query logs dataset (Pass et al., 2006). It did not add to the performance, most likely because any information it could provide was also provided by Google Books 2-grams. Similiarly, we tried extending the Google Books 2-grams data to include the first and third words of a selection of 3-grams, but this, too, appeared redundant with the 2-grams.

We also experimented with a feature based on bounding box annotations available in the Open-Images dataset (Krasin et al., 2017). We hoped it would help us capture attributes such as colors, materials, and shapes. While this feature did not improve the classifier's performance on the validation set, it did slightly improve the performance on the test set.

Before deciding on scikit-learn's LinearSVC,

Dataset	F1	Error (SEM)
train	.7617	± .0032
validation	.7281	± .0085
test	.7368	± .0091

Table 2: F_1 scores by dataset. The reported F_1 score is the arithmetic mean of the F_1 scores for both classes.

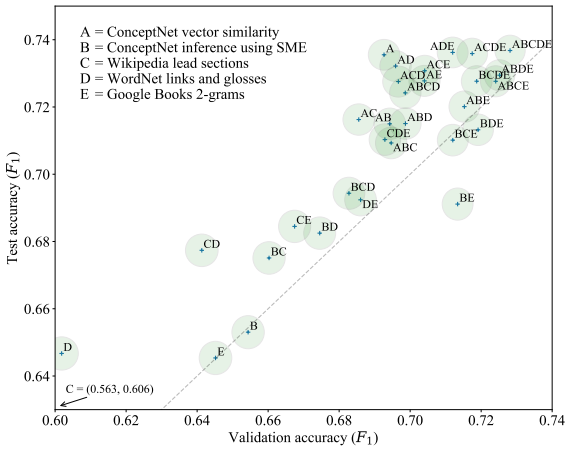

Figure 1: This ablation analysis shows the contributions of subsets of the five sources of features. Ellipses indicate standard error of the mean, assuming that the data is sampled from a larger, unseen set.

we experimented with a number of other classifiers. This included random forests, differentiable models made of multiple ReLU and sigmoid layers, and SVM with an RBF kernel or a polynomial kernel.

We also experimented with different parameters to LinearSVC, such as changing the default value of the penalty parameter C of the error term, changing the penalty from L_2 to L_1, solving the primal optimization problem instead of the dual problem, and changing the loss from squared hinge to hinge. These changes either led to lower performance or had no significant effect, so in the end we used LinearSVC with the default parameters for scikit-learn version 0.19.1.

5 Results

When trained on the training set, the classifier we describe achieved an F_1 score of 0.7617 on the training set, 0.7281 on the validation set, and 0.7368 on the test set. Table 2 shows these scores along with their standard error of the mean, supposing that these data sets were randomly sampled from larger sets.

5.1 Ablation Analysis

We performed an ablation analysis to see what the contribution of each of our five sources of features was. We evaluated classifiers that used all non-empty subsets of these sources. Figure 1 plots the results of these 31 classifiers when evaluated on the validation set and the test set.

It is likely that the classifier with all five sources (*ABCDE*) performed the best overall. It is in a statistical tie ($p > .05$) with *ABDE*, the classifier that omits Wikipedia as a source.

Most of the classifiers perfomed better on the test set than on the validation set, as shown by the dotted line. Some simple classifiers with very few features performed particularly well on the test set. One surprisingly high-performing classifier was *A* (ConceptNet vector similarity), which gets a test F_1 score of 0.7355 ± 0.0091. This is simple enough to be called a heuristic instead of a classifier, and we can express it in closed form. It is equivalent to this expression over ConceptNet Numberbatch embeddings:

$$\text{sim}(term_1, att) - \text{sim}(term_2, att) > 0.0961$$

where $\text{sim}(a, b) = \sqrt{\max\left(\frac{a \cdot b}{||a|| \cdot ||b||}, 0\right)}$.

It is interesting to note that source *A* (ConceptNet vector similarity) appears to dominate source *B* (ConceptNet SME) on the test data. SME led to improvements on the validation set, but on the test set, any classifier containing *AB* performs equal to or worse than the same classifier with *B* removed. This may indicate that the SME features were the most prone to overfitting, or that the validation set generally required making more difficult distinctions than the test set.

6 Reproducing These Results

The code for our classifier is available on GitHub at https://github.com/LuminosoInsight/semeval-discriminatt, and its input data is downloadable from https://zenodo.org/record/1183358.

References

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9(Aug):1871–1874.

Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. 2017. OpenImages: A public dataset for large-scale multi-label and multi-class image classification.

Yuri Lin, Jean-Baptiste Michel, Erez Lieberman Aiden, Jon Orwant, Will Brockman, and Slav Petrov. 2012. Syntactic annotations for the Google Books Ngram Corpus. In *Proceedings of the ACL 2012 system demonstrations*, pages 169–174. Association for Computational Linguistics.

George Miller, Christiane Fellbaum, Randee Tengi, P Wakefield, H Langone, and BR Haskell. 1998. *WordNet*. MIT Press Cambridge.

Denis Paperno, Alessandro Lenci, and Alicia Krebs. 2018. SemEval-2018 Task 10: Capturing discriminative attributes. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, United States. Association for Computational Linguistics.

Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *Proceedings of the 1st International Conference on Scalable Information Systems*, InfoScale '06, New York, NY, USA. ACM.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research*, 12(Oct):2825–2830.

Robert Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An open multilingual graph of general knowledge. In *AAAI*, San Francisco.

Robert Speer and Joanna Lowry-Duda. 2017. ConceptNet at SemEval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 85–89, Vancouver, Canada. Association for Computational Linguistics.

Wikipedia. 2017. Wikipedia, the free encyclopedia — English data export. (A collaborative project with thousands of authors.) Retrieved from `https://dumps.wikimedia.org/enwiki/` on 2017-12-20.