Using BERT for Distant Supervision

Utpal Mattoo (Email: utpal.mattoo@gmail.com)

Abstract

This report compares the approach taken in predicting entity relationships using multi-label regression with a transformer (BERT) based approach. Using BERT, each relationship is still predicted at a binary/logistic level as in multi-label regression, but instead of creating features using GloVe, POS tagging or a combination of various featurization methods (before the logistic step), this project proposes feature creation from text context using a BERT pre-processor followed by creation of BERT encodings for classification. BERT encodings can be used for prediction using a simple dense layered neural network, or a transformer decoder using the Attention mechanism. Length of the corpus text - middle context connecting the entity pairs (or other text fragments surrounding entities) could probably provide some intuition towards a choice between using Attention in a transformer decoder or a simple dense layered neural network.

1 Credits

This project was derived from the homework, lecture and utility code on Relationship Extraction and Distant Supervision. The starting point for the source code in this project came from the utility workbook at:

https://github.com/cgpotts/cs224u/blob/master/rel_ext_01_task.ipynb

This worbook was adapted to ideas for this project. The work was completed in Google Colaboratory

https://colab.research.google.com/

2 Introduction

Creating a rich feature set using a combination of bi-gram parts-of-speech for the middle context between entities, bag-of-words for middle context between entities and a bi-directional middle context between entities improves precision, recall and f-scores. The overall scores are better than using any one of these approaches independently. Refer to the section in related work for more detail on the background.

This paper explores featurization of the middle context (could also be the left/right context etc) of the entities in the corpus using the BERT pre-processor, encoder and decoder approach to come up with precision, recall and F-scores.

The challenges before using a complex modeling approach are related to data pre-processing:

(1) Pre-processing the middle context for the entity pair to remove blank or punctuation middles and correspondingly removing the True/False labels associated with the removed middle context to create (BERT feature sentence, labels).Blank or punctuation make little sense conceptually for vectorization.

(2) To understand if the middle context is even useful when it is a punctuation or blank/space symbol. Some distracting noise might make the model more robust to unseen (test) input, but punctuation or blank/space symbols don't appear to contain useful information. In other words, does having these actually improve precision, recall and F-scores or just add noise. In the multi-label logistic regression approach, these entity middles were all featurized and many of these even appeared in the top k middles for entity pairs. It's hard to explain why blank and punctuations symbols might be more prevalent for one relationship type versus another.

(3) If the middle context between entities is not long (so the vanishing gradients is not a problem, which is the reasoning behind not using a recurrent network), does it still make sense to use a transformer decoder or should we just use the existing approach and still do well?

3 Related Work

This project builds on research from three areas - which are joined in this one project: Feature generation (using lexical methods), relationship extraction and distance supervision and natural language inference.

For feature generation, this project looked at two papers. Zhou et al., 2007 [1] and Mintz, M. et.al [2]. In Zhou et al., 2007 [1], highly precise syntactic features were used to improve the performance in supervised information extraction using clean hand-labeled ACE data. This approach did not scale at that time because supervised training on a large corpus would mean creating precise large amounts of hand labeled training, validation and test data to maintain prediction quality. This approach could also be domain specific limiting application in a real-world setting, where the entire process of feature creation would have to be repeated for a different domain. Initially, this project report started down the path of exploring syntactic features to improve the performance of unsupervised or distantly supervised IE. Using a deep-learning approach might be more practical and broadly applicable because features would not be customized to a certain domain.

To address scalability of training Mintz, M. et.al [2] started with a smaller set of relations between entities (an initial bootstrap where the highly precise hand-engineered labeled set could prove useful) and then used Distant Supervision. Distant supervision here would mean the same as unsupervised information extraction (IE), weak supervision or self-supervision. Most previous research in bootstrapping or unsupervised IE has used only simple lexical features and this report has continued down the same path using a transformer (BERT) based approach. What if we just used large volumes of data using massive compute resources instead of created fine-tuned syntactic features for prediction.

For a broad overview of the problem refer to Gabor Angeli [3]. A deeper technical discussion of applying Relationship Extraction and Distant Supervision to Wikipedia can be found in Wu and Weld [4].

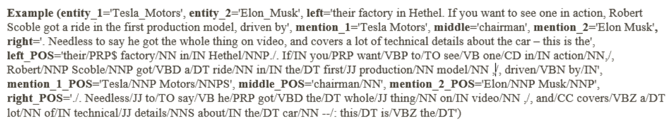
To introduce Natural Language Inference and recurrent models as alternatives to multi-label logistic regression for Distant Supervision, this project looked at research from Despina Christou et.al [5] and Christopher D. Manning et.al [6]. As described above in the section on introduction, this report implemented a BERT approach closest to [5].

4 Data

Two data sources are required:

a) Knowledge base (or KB): KB with 45,884 triples each consisting of a relation, a subject, and an object of the form (relation, subject, object). Ex: (worked at, Elon Musk, SpaceX). The KB used was already in a pre-processed form and initially and extracted from the Freebase Easy data dump. KB contains total 16 distinct relationships.

b) Corpus: This is the equivalent of free text, which is already structured to provide left/right/mid word context around the subject and object (the entity pair). This context helps derive the relationship. Corpus has with 331,696 examples and each example has the form:



Example (entity_1='Tesla_Motors', **entity_2**='Elon_Musk', **left**='their factory in Hethel. If you want to see one in action, Robert Scoble got a ride in the first production model, driven by', **mention_1**='Tesla Motors', **middle**='chairman', **mention_2**='Elon Musk', **right**='. Needless to say he got the whole thing on video, and covers a lot of technical details about the car – this is the', **left_POS**='their/PRP$ factory/NN in/IN Hethel/NNP./. If/IN you/PRP want/VBP to/TO see/VB one/CD in/IN action/NN,/, Robert/NNP Scoble/NNP got/VBD a/DT ride/NN in/IN the/DT first/JJ production/NN model/NN ,/, driven/VBN by/IN', **mention_1_POS**='Tesla/NNP Motors/NNPS', **middle_POS**='chairman/NN', **mention_2_POS**='Elon/NNP Musk/NNP', **right_POS**='./. Needless/JJ to/TO say/VB he/PRP got/VBD the/DT whole/JJ thing/NN on/IN video/NN ,/, and/CC covers/VBZ a/DT lot/NN of/IN technical/JJ details/NNS about/IN the/DT car/NN --/: this/DT is/VBZ the/DT')

Fig 1: Example corpus tuple for an entity pair

The last five fields contain the same five chunks of text, but this time annotated with part-of-speech (POS) tags, which may turn out to be useful when we start building models for relation extraction. The corpus we'll use for this project is derived from the Wikilink article: https://code.google.com/archive/p/wiki-links/ announced by Google in 2013: https://ai.googleblog.com/2013/03/learning-from-big-data-40-million.html. This dataset contains over 40M mentions of 3M distinct entities spanning 10M webpages. It provides entity resolutions by mapping each entity mention to a Wikipedia URL. In order to do relation extraction, we need pairs of entity mentions, and we need context around and between the two mentions. UMass has provided an expanded version of Wikilinks http://www.iesl.cs.umass.edu/data/data-wiki-links which includes the context around each entity mention. Pairs of entities along with their contexts in stitched together form are used as the corpus.

5 Baseline results

Fig 2 shows shows problems which we can still solve (marked with an 'x'). The focus of this project is at the intersection of the (first row, last column): using lexical features with a Transformer.

Fig 3 and Fig 4 show precision, recall and F-scores obtained when featurizers are combined before prediction.

In Fig 4. combining featurizers 2, 3, 4 - with total features now at 71274 for the per-relation logistic regression, led to a large increase in Recall and F-score (Precision improved very slightly or stayed about same). For cases where we used multiple featurizers together (1 and 5), the improvement in precision over individual featurizer cases (2 and 3) was very little. But Recall and F-scores improved by a larger percentage.

If GloVe alone were used as a Vectorizer for middle features (words between entities) and then a multi-label logistic regression was run on the feature vectors, we would see the results in Fig 5 (all scores for each relation are shown for GloVe. Only aggregate or macro numbers are reported in Fig 3 and Fig 4 for featurizer cases 1, 2, 3, 4, 5).

6 Model

BERT versions included below were used for modeling the middle context. Choice of a small BERT implementation made sense because it is useful for smaller architectures. Following the initial data pre-processing and formatting (the more complex part), we go through a BERT pre-processing and an actual BERT encoder step. Once BERT encodings are ready, this project uses a drop out rate of 0.1 for over-fitting and a single output dense layer for binary prediction with a sigmoid non-linearity. BERT pre-processor:

https://tfhub.dev/tensorflow/bert_en_uncased_preprocess/3

BERT Encoder

https://tfhub.dev/tensorflow/small_bert/bert_en_uncased_L-2_H-128_A-2/1

Included source code uses utility (custom library) code, google colaboratory set up, data pre-processing for structuring data into an expected format for BERT pre-processing and a small BERT model.

While there are 16 possible relationships between entity pairs, this project builds and test the model for one relationship for ease of testing. With little change, this code can be adapted to cover all 16 relationships. The key challenge in the modeling step has been the creation of negative examples for training and this is a key area of continued work, but needs some input on the best way to proceed. My recommendation for creating negative training examples is included in the Analysis section. Standard choices for learning rate, number of epochs were made without using hyper-parameter selection.

7 Experiments

Some code changes could could be called experiments to then inform data and model quality.

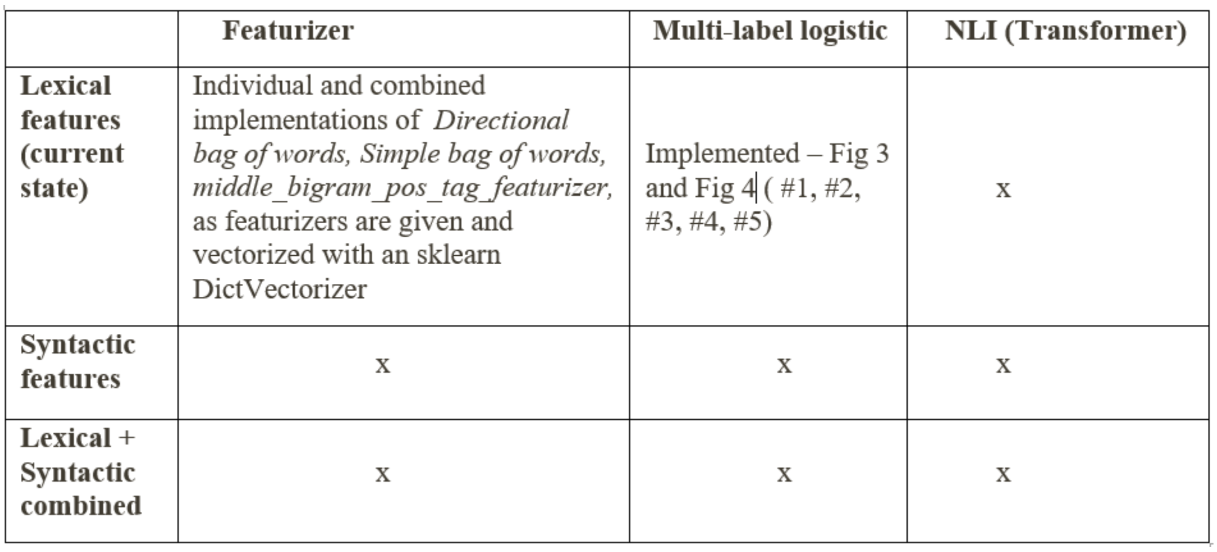
First, for a given relationship and entity pair, I was able to extend the framework in the method evaluate() (included in the submitted code file) to collect a list of top k mids and mids between 0 and k. The length and content (punctuation vs standard english alphabet, for example) could then be related to the precision, recall and F-scores. We could find ways to justify/refute (or remain undecided) about how length and content matter to metrics. This might inform training data creation, data cleansing or even model choice. Does using a transfomer model with attention even make sense from a technical complexity or compute budget perspective.

A second experiment, in the method evaluate() relates to code changes for capturing which knowledgebase tuples have corresponding examples in the Corpus. This could also then inform our understanding of the quality training data (knowledgebase and Corpus) and relate this understanding to the metrics we create from the model. We want to know: is our problem the data, how we pre-processed it or is it the modeling approach?

8 Analysis

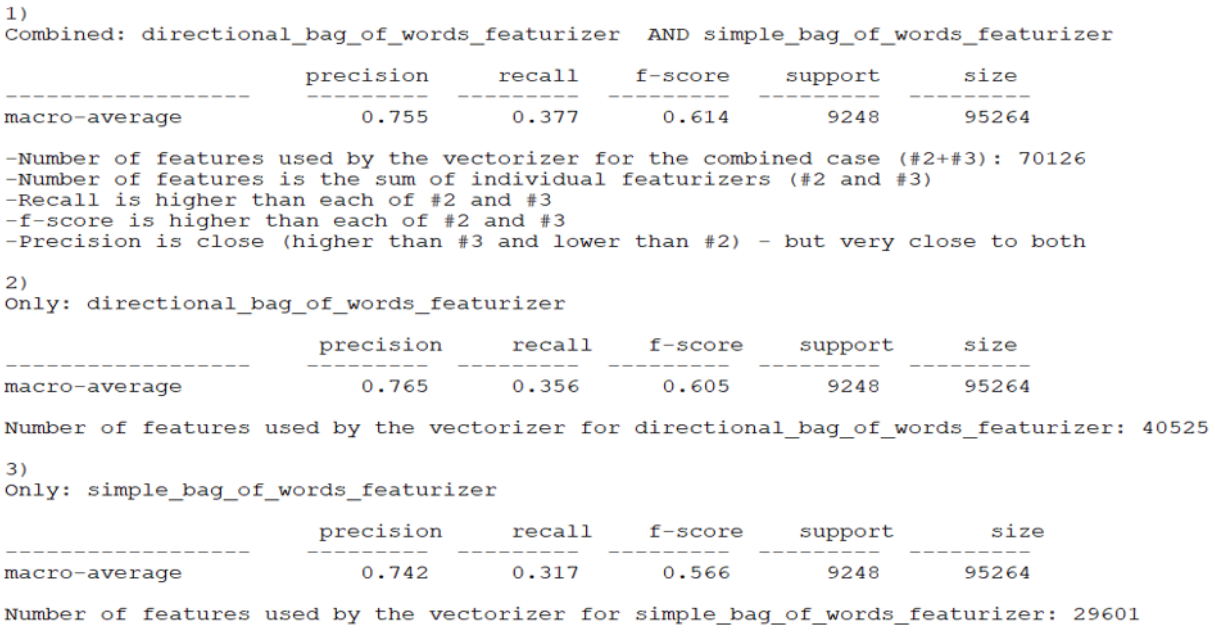
The True/False labels in the framework code are based on related and un-related entities – by only considering the Knowledgebase tuples. An entity pair is considered related in the context of a given relationship, if the two entities and the relationship are attributes in the knowledgebase tuple. An entity pair are un-related if they don't form an entity pair in any knowledgebase tuple for any relationship.

For purposes of classification, where we need the (contextual text, Label) for input to BERT – for both positive and negative example, it is obvious that we would assign 'True' for the Label for a given (entity pair, relationship) when a corpus example is found for the Knowledgebase Tuple.



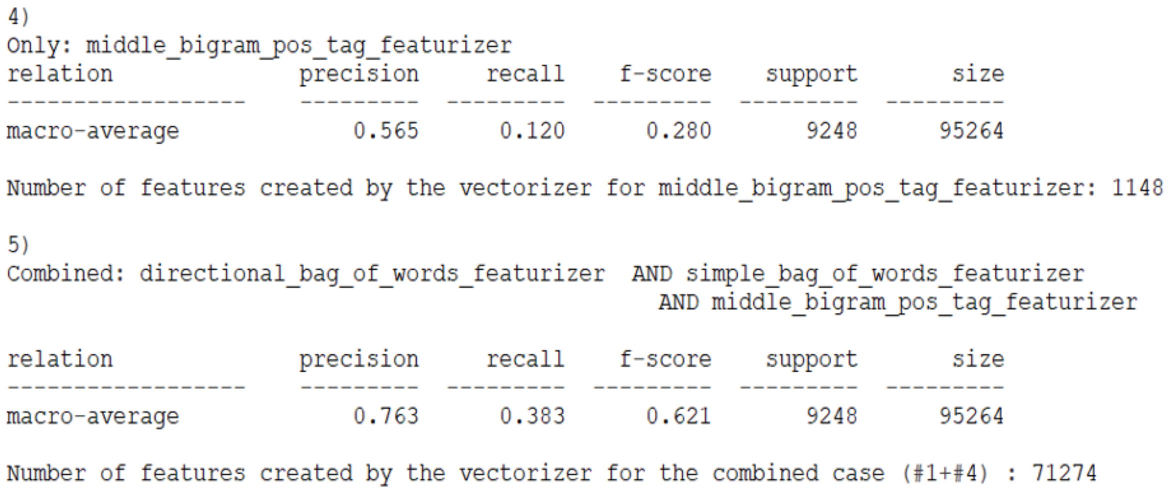
	Featurizer	Multi-label logistic	NLI (Transformer)
Lexical features (current state)	Individual and combined implementations of *Directional bag of words, Simple bag of words, middle_bigram_pos_tag_featurizer,* as featurizers are given and vectorized with an sklearn DictVectorizer	Implemented – Fig 3 and Fig 4 (#1, #2, #3, #4, #5)	x
Syntactic features	x	x	x
Lexical + Syntactic combined	x	x	x

Fig 2: Current state and work choices



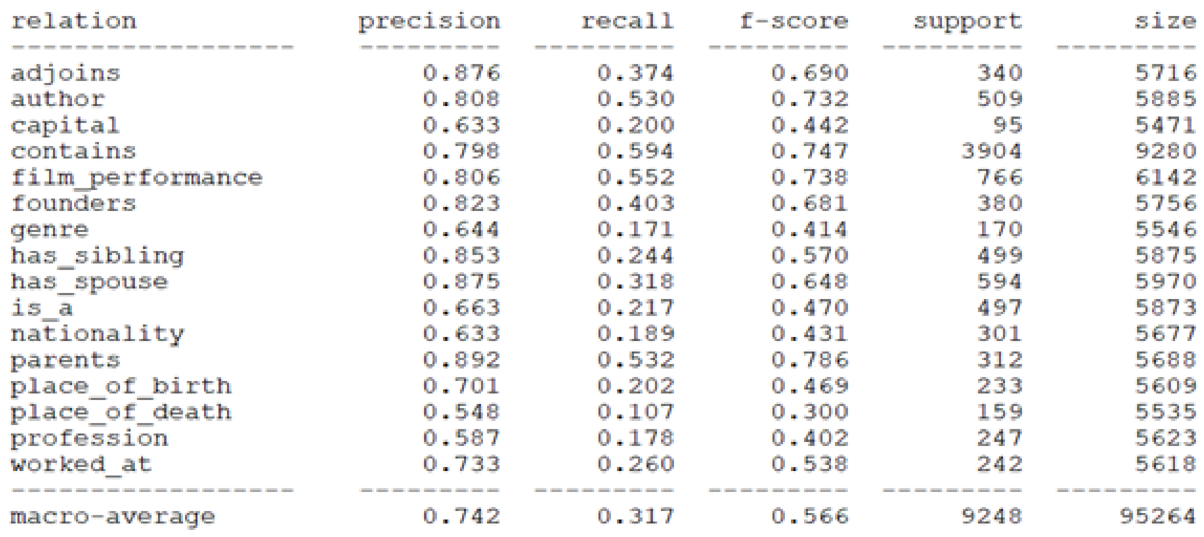
```
1)
Combined: directional_bag_of_words_featurizer  AND simple_bag_of_words_featurizer

                           precision     recall    f-score    support      size
------------------         ---------  ---------  ---------  ---------  ---------
macro-average                  0.755      0.377      0.614       9248      95264

-Number of features used by the vectorizer for the combined case (#2+#3): 70126
-Number of features is the sum of individual featurizers (#2 and #3)
-Recall is higher than each of #2 and #3
-f-score is higher than each of #2 and #3
-Precision is close (higher than #3 and lower than #2) - but very close to both

2)
Only: directional_bag_of_words_featurizer

                            precision     recall    f-score    support      size
------------------         ---------  ---------  ---------  ---------  ---------
macro-average                  0.765      0.356      0.605       9248      95264

Number of features used by the vectorizer for directional_bag_of_words_featurizer: 40525

3)
Only: simple_bag_of_words_featurizer

                            precision     recall    f-score    support       size
------------------         ---------  ---------  ---------  ---------  ---------
macro-average                  0.742      0.317      0.566       9248      95264

Number of features used by the vectorizer for simple_bag_of_words_featurizer: 29601
```

Fig 3: Combining featurizers



```
4)
Only: middle_bigram_pos_tag_featurizer
relation              precision     recall    f-score    support       size
------------------    ---------  ---------  ---------  ---------  ---------
macro-average             0.565      0.120      0.280       9248      95264

Number of features created by the vectorizer for middle_bigram_pos_tag_featurizer: 1148

5)
Combined: directional_bag_of_words_featurizer  AND simple_bag_of_words_featurizer
                                                AND middle_bigram_pos_tag_featurizer

relation              precision     recall    f-score    support       size
------------------    ---------  ---------  ---------  ---------  ---------
macro-average             0.763      0.383      0.621       9248      95264

Number of features created by the vectorizer for the combined case (#1+#4) : 71274
```

Fig 4: Gains with an additional featurizer



```
relation             precision    recall   f-score   support      size
------------------   ---------  --------- ---------  ---------  ---------
adjoins                  0.876     0.374     0.690       340       5716
author                   0.808     0.530     0.732       509       5885
capital                  0.633     0.200     0.442        95       5471
contains                 0.798     0.594     0.747      3904       9280
film_performance         0.806     0.552     0.738       766       6142
founders                 0.823     0.403     0.681       380       5756
genre                    0.644     0.171     0.414       170       5546
has_sibling              0.853     0.244     0.570       499       5875
has_spouse               0.875     0.318     0.648       594       5970
is_a                     0.663     0.217     0.470       497       5873
nationality              0.633     0.189     0.431       301       5677
parents                  0.892     0.532     0.786       312       5688
place_of_birth           0.701     0.202     0.469       233       5609
place_of_death           0.548     0.107     0.300       159       5535
profession               0.587     0.178     0.402       247       5623
worked_at                0.733     0.260     0.538       242       5618
------------------   ---------  --------- ---------  ---------  ---------
macro-average            0.742     0.317     0.566      9248      95264
```

Fig 5: Per relation and scores with GloVe

When we don't have the text context because the entity pair for the given relationship in the Knowledgebase was not found in the Corpus, we could generate a middle sentence for a random entity pair for any other relationship, and use the contextual text from that randomly chosen corpus example and because this is an unrelated entity pair than the one we started with, we will assign a relationship label of False. This could be used as a negative training example. BERT will need both – the negative label and corresponding text for the false case. As to how many negative training examples could be chosen for a Knowledgebase Triple, that remains an open question.

As for when during pre-processing to create negative training examples, there could be two possible stages - during featurization or before any featurization begins. For during featurization, if Corpus does not contain any contextual text for the entities in the knowledgebase, then we could assign a label 'False' to indicate that the entities are not related (even though if the corpus were big enough, we could have found contextual text in the corpus). So, we have the negative value for the Label, but since we did not find the negative contextual example in the corpus, we will use the context text from an unrelated relationship to serve as a negative example. We could find the top k mid (or the left/right) from the random corpus example to serve as a strong feature for the negative case.

Alternatively, during initial processing, we can upfront look for related entity pairs (for a given relationship) in the Knowledgebase that have no contextual text in the Corpus. For these pairs, we can assign a 'False' label and use contextual text from a different relationship. In this case, we know upfront that we have negative and positive examples and do not need to create these in the middle of processing (or during featurization).

9 Conclusion

When seen in context of the original goal of improving multi-label regression using a transformer (BERT), this project is currently a work in progress. I am interested in removing punctuation's, small length mids and remove corresponding labels before a modeling step to derive a meaningful conclusion. Past this step, it could make sense to classify and derive metrics. However, the following conclusions can be drawn:

1. We could consider the development of a framework for generating negative (sentence, Label) tuples when Knowledgebase tuples are not found in the Corpus. Comparing unrelated entities while only looking within the Knowledgebase and not cross matching between Knowledgebase and Corpus means we don't explore ways to make training data complete. After this framework is complete (we have both negative and positive inputs), we can then dive deeper into encoding with BERT the output of the BERT pre-processor.

2. Capture length (k) and quality of the middle context of each entity pair for each relationship. These can also then be seen side-by-side with classification results to understand how training data quality affects results.

10 Required Authorship statement

This work was done independently.

11 References

1. Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In ACL-05, pages 427–434, Ann Arbor, MI;

https://www.aclweb.org/anthology/P05-1053.pdf

2. Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2, 1003–1011. Assoc. for Comp. Linguistics.

https://www.aclweb.org/anthology/P09-1113

3. Gabor Angeli, Victor Zhong, Danqi Chen, Arun Chaganty, Jason Bolton, Melvin Johnson Premkumar, Panupong Pasupat, Sonal Gupta, Chris Manning; Bootstrapped Self Training for Knowledge Base Population Text Analysis Conference Proceedings. 2015

https://cs.stanford.edu/~angeli/papers/2016-tac-kbp.pdf

4. Fei Wu and Daniel S. Weld. 2007. Autonomously semantifying wikipedia. In CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pages 41–50, Lisbon, Portugal

http://turing.cs.washington.edu/papers/cikm07.pdf

5. Despina Christou, Grigorios Tsoumakas. Improving Distantly-Supervised Relation Extraction through BERT-based Label Instance Embeddings

https://arxiv.org/abs/2102.01156

6. Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, Christopher D. Manning; Position-aware Attention and Supervised Data Improve Slot Filling;

https://cs.stanford.edu/~angeli/papers/2017-emnlp-tacred.pdf