Review of Current Text Representation Technics for Sematic Relationship Extraction

M. GAŁUSZA

michal.galusza@wat.edu.pl

Military University of Technology, Faculty of Cybernetics
Institute of Computer and Information Systems
Kaliskiego St. 2, 00-908 Warsaw, Poland

Article provides review on current most popular text processing technics; sketches their evolution and compares sequence and dependency models in detecting semantic relationship between words.

Keywords: NLP, Relationship Extraction, Context Dependency Graph

DOI: 10.5604/01.3001.0015.2733

1. Introduction

Natural Language Processing (NLP) aims to extract information from text. As it is impossible to consider text as data directly to simply "read the text", a large set of processing methods have been developed to transform text to a representation that allows inference.

Relationship Extraction (RE) is a cornerstone of applications requiring relational understanding of unstructured text such as question answering, knowledge base population or biomedical knowledge discovery.

Nowadays, NLP relies mostly on sequence-based methods which, in order to efficiently represent the text, use preprocessing together with large corpus. In most cases text preprocessing is limited to some standards which include lemmatization, stop words/punctation removal, part of speech tagging.

More advanced tools (i.e. Spacy [24]) offer coreference resolution, other (i.e.: Gensim [21]) bigram detection where two words expression like i.e.: New York are considered a single token New York while i.e.: "a new dress" remains "a new dress". The state-of-the--art models: BERT [18], GPT [17], [19], ELMO [20] do not preprocess text at all relaying solely on statistics of words co-occurrence to construct representation of the text.

As long as corpus is large enough to be abundant in varying usage of words, statistics may provide meaningful insides into language patterns and allow reasoning about relationships and semantics of words.

Large corpus or large annotated training set, however, in specialized cases is prohibitively little as it is difficult to collect a large corpus or large training set to detect i.e.: root causes of dam failures and resulting floods. In such a case, distilling as much data as possible from text is mandatory in order to reason about its semantics.

One of the solutions is to reconstruct complete grammatical dependency – Dependency Tree or Dependency Graph – from the sentences and represent the sentence or complete passage (context) as directed (attributed) acyclic graph (DAG) $G = (V, E)$ where V – vertices are words with attributes i.e.: POS and E – edges is a labelled grammatical relationship between vertices: head word and modifier word.

2. Sequence Based Methods

In order to put Dependency Tree Methods in a context, I will sketch the most popular NLP modeling approach nowadays.

Sequence Based Methods (SBM) originate from statistical model of language where the model estimates the probability of word given the defined context:

$$\hat{P}(w_1^T) = \prod_{t=1}^{T} \hat{P}(w_t | w_1^{t-1}) \qquad (1)$$

where w_t is the t-th word of the context of length T.

Although the intuition is straightforward, namely, the closer the word the impact to distribution w_t is bigger, due to curse of dimensionality of the language direct estimation of the probability is not feasible. Early and effective computational methods used n-gram approximation of defined order. Seminal paper of Y. Bengio [12] uses recurrent neural network to learn the probability function given the context words:

$$\hat{P}(w_t|w_{t-1},\cdots w_{t-n+1}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \tag{2}$$

where y_{w_t} is an unnormalized log-probability – the output of the network.

However, training the network although possible remained computationally expensive due to the Softmax form of the estimated probability function – one has to evaluate function value of the word given all words in the vocabulary (regardless if in or not in the context).

Another issue is that language model is not linear, meaning that words further away in the context may have bigger impact on the distribution of w_t than words closer in the context. A Noise Contrastive Estimation (NCE) as proposed in word2vec'c Skip-Gram [16] solves the problem of effective estimation of log-probability. LSTM [26] and Attention model [23] solves the problem of non-linear influence. The state-of-the-art models BERT [18] and GPT2 [17], [19] through Transformer architecture [22] extend the attention model [23] while still relying on large corpus i.e.: Internet Common Crawl.

3. Context and Preprocessing

Preprocessing is essential in SBMs to reduce a "noise" in text by removing i.e.: words that appear frequently in many contexts, hence conveying no significant information. Selection of preprocessing methods ([13]), may impact the quality of results and as of now, there is no "golden rule" how to use preprocessing apart from one that the more data is available the less preprocessing impacts the overall quality of the model.

A context is defined as a fixed length window of size (number of words) T before and after the target word. Example: "Paris is a nice city in France and it is well-known for its museums." if T = 3 and target word is "France", then its context contains words: "nice", "city", "in", "it", "is", "well-known". It is now obvious that selection of parameter T is crucial as for T = 3, relationship between Paris – France and Paris – museums, in this context, is lost. T remains a parameter which is set heuristically.

4. Word2Vec

Word2vec [15], [16] is an example of SBM algorithm and because of its word embedding, is very successful NLP technic, yet relying on large corpus and posing some difficulties to reason about semantics of words.

Direct use of word2vec will not allow to i.e.: disambiguate word sense as the semantics of word is averaged over the corpus. For example, the embedding of the word "bank" will remain the same in the following sentences "After robbing the bank the burglar decided to take some rest. He sat on the bank of the river and had a snack."

Original word2vec does a little preprocessing namely: punctation, colons and full stops, are removed. Apostrophes denoting genitive form i.e.: "Peter's", hyphens denoting multiword expressions i.e.: "a well-being" are remained. Sentences are lowercased. Example: "Paris is a nice city in France, and it is well--known for its museums." will be converted to "paris is a nice city in france and it is well--known for its museums". For a Skip-Gram algorithm, window size T is set to 10 so relationship between "Paris", "France" and "museums" would be captured.

Additionally, very rare words, appearing less than 5 times in a corpus, are removed. Very frequent words (stop words) are not removed but their NCE probability is damped.

5. Bi-directional Encoder Representation from Transformers – BERT

Transformer architecture [22] is currently state-of-the-art in solving NLP related tasks. Although there are variations in ways how Transformer architecture is implemented and trained (i.e., GPT vs BERT) simplified preprocessing and large corpus remains a common part of all.

Specifically, BERT [18] uses Masked Language Model (MLM) together with next sentence prediction task.

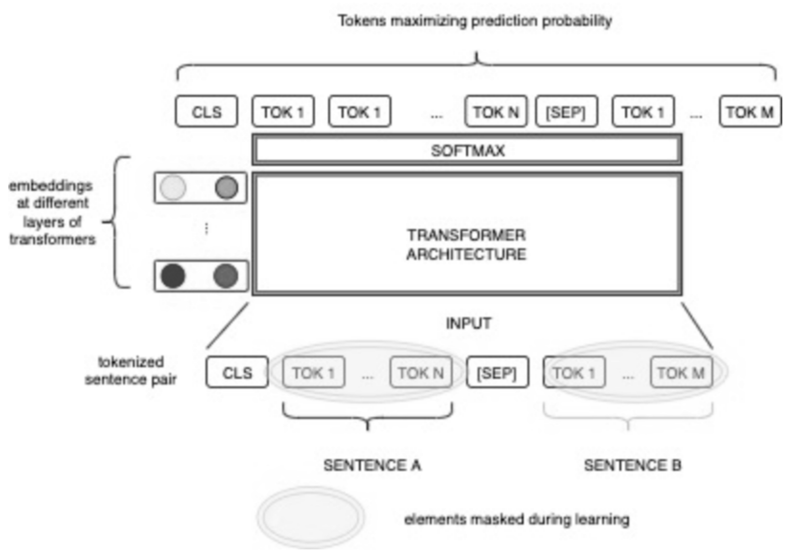


Fig. 1. BERT Architecture

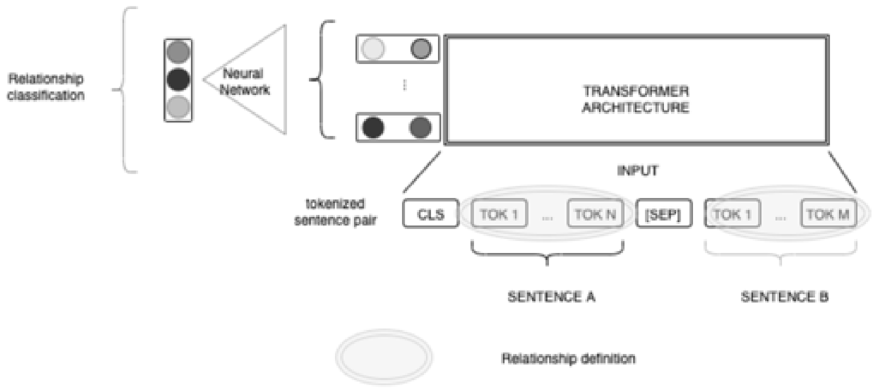


Fig. 2. Transfer Learning Architecture

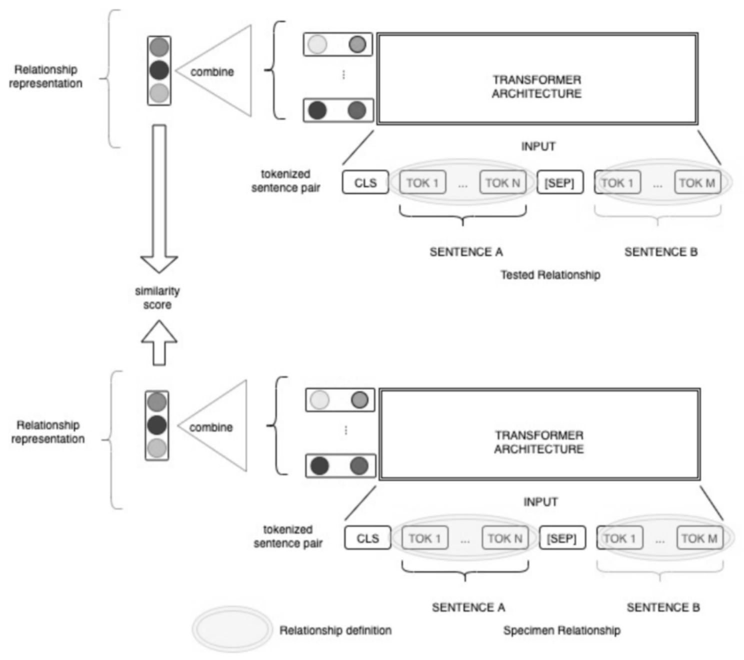


Fig. 3. Meta-Learning Architecture

Preprocessing of the corpus is limited to wordpiece tokenization ([18]), lowercasing and sentence punctation removal.

Context is a limited to 512 wordpiece tokens within two-sentence window. Effectiveness of BERT is a result of combining context-word prediction with next sentence prediction task, large in terms of parameters bi--directional Transformer architecture and huge Common Crawl corpus.

As the result, BERT's word embedding is contextualized which means that, in contrast to word2vec, it will disambiguate word sense.

The embedding of the word "bank" will be different in the following sentences "After robbing the bank the burglar decided to take some rest. He sat on the bank of the river and had a snack."

The common approach to use BERT (or other Transformer based models) to solve NLP task i.e. Relationship Extraction, is via Transfer Learning ([14]), Figure 2, where on top of the Language Model, additional classifier is trained on large set of examples, usually manually annotated, to detect features in the text. Another, yet not very effective, approach is a meta (few-shot) learning ([14]). Meta-learning relies on handful of examples to reason about the text. Retraining (adaptation) of the transformer model to the task – as in Transfer Learning – is replaced by similarity between examples and candidates as presented on Figure 3.

6. Dependency based methods (DBM)

None of the methods presented in SBM tries to augment the context with syntactical information.

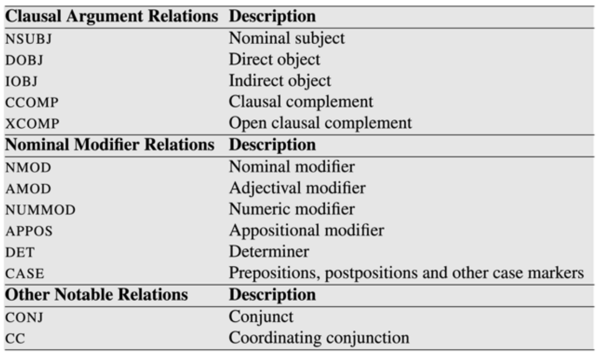
Following section will explain how this information is available and how inference on text can benefit from it in NLP tasks.

7. A (Sentence) Dependency Tree

A Dependency Tree (DT) is a directed grammatical relation between two (head and dependent) words in a sentence. In addition to linking a pair of words, a relation determines grammatical function which connects them.

Complete repository of dependency relations that are linguistically motivated and computationally useful is called Universal Dependencies (UD). Details of UD are out of scope, however, relations fall into two general categories:

- clausal relations that are usually linked to a verb i.e.: to cancel flight, to break a dam.
- modifier relations that categorize words indicated by the head i.e.: concrete dam, late flight.



Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

Fig. 4. UD Basic Relationship Types

The structure of the sentence "Paris is a nice city in France, and it is well-known for its museums." shall be described as a dependency tree (Figure 5).

It is now clear that distance between Paris – France and Paris – museums is identical in terms of graph distance and we can apply graph mining technics to derive semantic relations between them.

8. Transition Sequence Algorithm Shift-Reduce Parsing

Although being an addition to text processing, DT algorithms are efficient computationally and combine classic stack-based approach, originally developed to analyze programming languages, with neural classification task detecting relation type between words. The Shift-Reduce Parsing ([2], [3], [4]) has complexity O(n) with respect to words in the sentence hence the overhead is reasonable.

Processing scheme (Figure 6) consists of following steps:

1. An input buffer which holds words in the sentence read from left to right.
2. A stack which holds words for processing.
3. A parser consists of set of actions:
   - Left arc: asserts head-dependent relationship between top-of-the-stack word and second top-of-the-stack one and removes second word from the stack;
   - Right arc: asserts relationship head--dependent between second top-of-the--stack and top-of-the-stack word and removes top word from the stack;
   - Shift: removes word from buffer and puts it on stack.

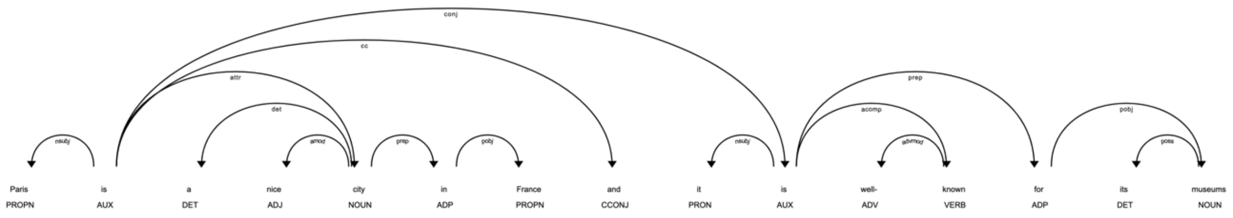


Fig. 5. Sentence Dependecy Tree Example

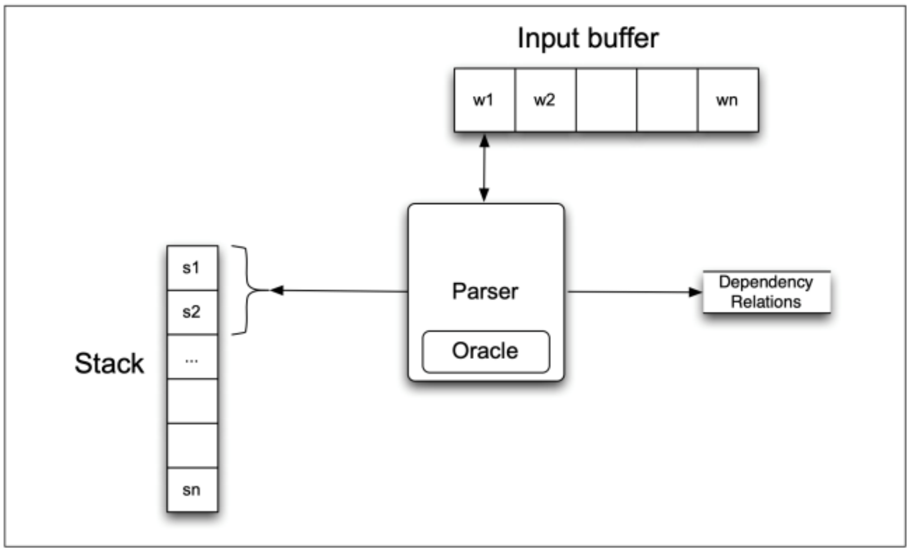


Fig. 6. Shift-Reduce Parsing Architecture

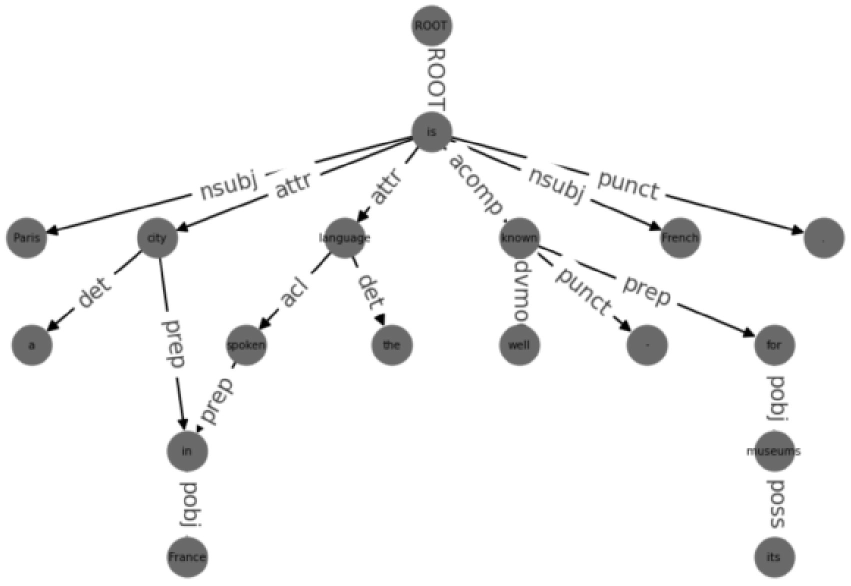


Fig. 7. Context Graph Example

Parsing algorithm efficiently decomposes the sentence detecting left or right arcs, however it does not detect the type of arcs as defined in UD. The *Oracle* is responsible for detection of edge type. This is achieved via trained multicategory classifier that takes the state of the stack: top, second top word and predicted relations so far to predict the relation type on the current arc.

The classifier is trained in supervised manner using existing annotated treebanks (i.e.: Penn Treebank). Various algorithms and various models are used i.e.: Deep Learning, Graph-Based or feature-based [1], [4].

Combination of Shift-Reduce Parsing and *the Oracle* reconstructs grammatical structure of a sentence.

9. Context Dependency Graph (CDG)

DT models syntactic structure of the sentence. As inferring semantics of the word may require context of several sentences, a mechanism to represent the passage – or larger context – as a dependency graph is required.

A successful implementation has been published in [5]. The algorithm to construct a CDG, which is a graph G = (V, E) is straightforward, each node V in the graph G represents word and its POS, each edge E represents dependency relation in the sentence in the context. Nodes and edges are inserted only once.

Example context sentences:
"Paris is a city in France",
"Paris is well-known for its museums",
"French is the language spoken in France".

Graph representation of the context (Figure 7) allows modeling of local and non-local dependencies between words and enables graph algorithms to extract them from text. It is now possible i.e.: to assert that French is spoken in Paris by detecting a relationship between nodes "Paris" and "French".

Relationship Extraction (RE) is one of the NLP tasks which aims to extract sematic relatedness of words in the context i.e.: in the sentence "Pierre lives in Paris and works for Société Générale." semantic relationship R between subject: "Pierre" and object: "Société Générale" is PLACE_OF_WORK identified by a predicate "works". RE is possible in SBM via Transfer Learning or Meta-Learning, however, in many cases, either annotated training examples or corpus large enough to train SBM model is not available. DBMs are complementary to SBMs. Following, I will review a handful of representatives of RE algorithms based on DT.

10. Dependency Tree Kernel

Tree kernels [26], [27] is the first method to use kernel to work directly on sentence parse trees and it is the first method to take other than feature engineering approach in NLP.

In machine learning, kernel function is a similarity function between objects which is then used in a supervised setup in i.e.: Support Vector Machine (SVM) [28].

Taking the example "John Smith is the chief of the Hardcom Corporation.". [26] we shall detect the relationship in the sentence as follows:

1. Given Named Entity Recognition [24] we assume that "John Smith" is a Person, and "Hardcom Corporation" is an Organization.
2. Given sentence dependency tree, we will construct a least common subtree linking Person and Organization.
3. Because objects we link is a Person and an Organization, we assume that relationship linking them is "affiliation". We state that the subtree represents "affiliation" relationship.
4. We will then confirm or reject the hypothesis by constructing a subtree kernel [26], [27] and running SVM detection trained on annotated examples.

11. Shortest Path Kernels

Shortest Path (SP) Kernels [6] infer relationship R between words assuming that it is present on the shortest path linking two words in the undirected version of the DT.

Relationship R will exist if the predicate will link both words directly or through preposition. Structure of the SP is used to construct a kernel which is then plugged in SVM algorithm to detect type of the relationship based on a training set.

Kernel is defined as:

$$K(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} 0, & m \neq n \\ \prod_{i=1}^{n} c(x_i, y_i), & m = n \end{cases} \tag{3}$$

where: $c(x_i, y_i) = |x_i \cap y_i|$ is a number of common word classes between x and y in the path. If paths have different length K = 0 by definition.

Example:

SP x: "his actions in Brcko"

SP y: "his arrival in Beijing"

Both SP's are examples of AT_LOCATION relationship with predicate "arrival" and "actions". SP y is assumed to be a training example.

Kernel representation for both SP's:

1. $x = [x_1\, x_2\, x_3\, x_4\, x_5\, x_6\, x_7]$

where:

x_1 = {his, PRP, PERSON},
x_2 = {→},
x_3 = {actions, NNS, Noun},
x_4 = {←},
x_5 = {in, IN},
x_6 = {←},
x_7 = {Brcko, NNP, Noun, LOCATION}

2. $y = [y_1\, y_2\, y_3\, y_4\, y_5\, y_6\, y_7]$

where:

y_1 = {his, PRP, PERSON},
y_2 = {→},
y_3 = {arrival, NN, Noun},
y_4 = {←},
y_5 = {in, IN},
y_6 = {←},
y_7 = {Beijing, NNP, Noun, LOCATION}

where PERSON, LOCATION are attributes augmented by NER algorithm ([24]). Based on the definition, value of the kernel: $K(x, y) = 3\ x\ 1\ x\ 1\ x\ 1\ x\ 2\ x\ 3 = 18$.

12. Graph Convolution over Pruned Dependency Tree

In contrast to kernel methods, graph deep learning methods do not suffer from sparse feature space but pool information over arbitrary dependency structure via convolution over neighboring nodes (words). Latest development in Deep Learning specifically for graphs: Graph Convolution Networks – GCN [9], GraphSAGE – [11] and Graph Attention Network – GAT [10] impacted RE state-of--the-art.

Pruned Dependency Tree approach [8] is based on the assumption that models using Shortest Path (SP) on undirected DT, will lose essential semantic information (i.e.: negation) which are usually excluded from SP. Exemplary sentence: "He was not a relative of Mike Cane" will generate SP between "He" and "Cane": $he - was - relative - of - Cane$ leaving information that in fact he was not.

Pruning must then extend SP with additional nodes K distance away from SP. According to [8] K = 1 yields best results in balancing relevant vs irrelevant information in the DT.

An undirected, pruned dependency tree between an object "Cane" and a subject "He" of the relationship is then fed through GCN to pool contextual information and then to classify relationship type between them. The pruned DT for a sentence created GCN convolution is emphasized on the right of Figure 9.

Separate embeddings for subject h_s, object h_o and connecting tree h_{sent} are concatenated. On top of concatenated embeddings, a feedforward neural network with Softmax output is trained to estimate probability distribution across a dictionary of relations.

13. Attention Guided Graph Convolutional Networks for Relation Extraction

Attention mechanism is crucial in effectiveness of Transformer architecture. It learns the impact of a words in the context on the meaning of given word.

Graph attention is a mechanism that learns an impact of a nodes from the neighborhood on the classification of the current node. Attention mechanism improved classification tasks state-of-the-art in graph domain [10].

In DT context, attention can be viewed as a "soft pruning" strategy [7] and comparing to a fixed K = 1 pruning strategy, it learns parameter K effectively using multi-head attention [23]. Key elements (Figure 10) of the algorithm are:

1, Representation of CGD or DT, neighborhood matrix A, is replaced by a fully connected weighted graph with neighborhood matrix $\tilde{A}$.
2. Attention Layer which will estimate parameters of matrix $\tilde{A}$ to allow to attend words several hops away (K > 1) from each other.
3. A Dense Layer (GCN) that will convolute the neighborhood given $\tilde{A}$ and produce final node embedding that will reflect the attention.
4. A feedforward neural network to classify relationship.

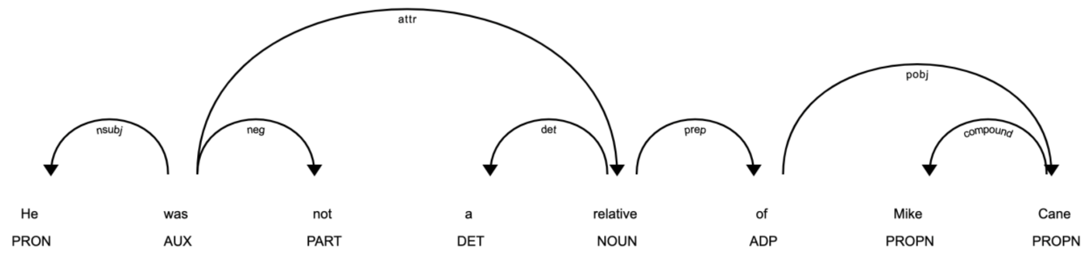


Fig. 8. Sentence Dependency Tree

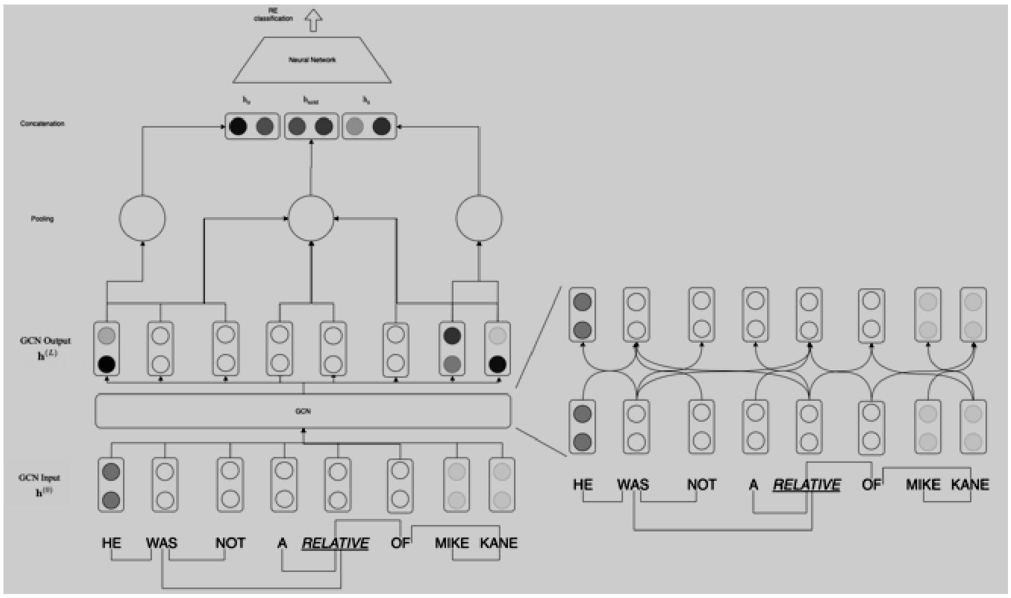


Fig. 9. GC over Pruned DT Architecture

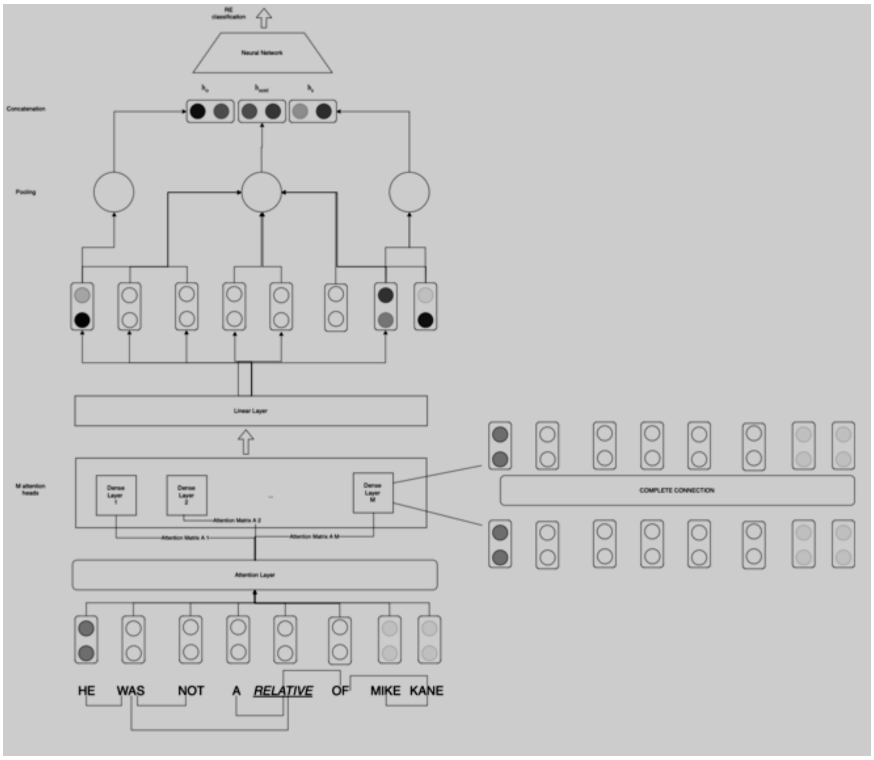


Fig. 10. Attention Driven GCN Architecture

In order to attend influence of several words at the same time multi-head attention is required hence:

1. Several matrixes $\tilde{A}$ are needed – one for each head.
2. Several dense layers to convolute neighborhoods for each attention driven matrix $\tilde{A}$ each producing embedding for each attention head.
3. A linear layer that will aggregate each embedding producing single one.

14. Summary

Dependency based approach to represent text is interesting for following reasons:

- It utilizes dependency parsing to construct a grammatical structure and therefore increase information included in a text.
- It provides a representation of text that is a basis for a large set of algorithms from graph mining and graph deep learning domains
- It does not require significant corpus to run inference
- It is on par with SBM for state-of-the-art algorithms. Both SBM and DBM use Attention in their best performing models.

15. Acknowledgements

Author was supported by Amazon Compute grant PS_R_POC_FY2020_Q1_MUT_Galusza_5000 to run simulations to prepare this article.

16. Bibliography

[1] Nivre J., “Deep Contextualized Word Embeddings in Transition-Based and Graph-Based Dependency Parsing – A Tale of Two Parsers Revisited”, in: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processingand the 9th International Joint Conference on Natural Language Processing*, 2755–2768, Hong Kong, China, November 3–7, 2019.

[2] Nivre J., “Incrementality in deterministic dependency parsing”, in: *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, 50–57, July 2004.

[3] Nivre J., “An efficient algorithm for projective dependency parsing”, in: *Proceedings of the Eighth International Conference on Parsing Technologies*, 149–160, Nancy, France, 2003.

[4] Jurafsky D., “Speech and Language Processing”, https://web.stanford.edu/~jurafsky/slp3/15.pdf.

[5] Franciscus N., Xuguang Ren, Stantic B., “Dependency graph for short text extraction and summarization”, *Journal of Information and Telecommunication*, Vol. 3, No. 4, 413–429 (2019).

[6] Bunescu R., Mooney R., “A Shortest Path Dependency Kernel for Relation Extraction”, in: *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 724–731, Vancouver, British Columbia, Canada, 2005.

[7] Guo Z., Zhang Y., Lu W., “Attention Guided Graph Convolutional Networks for Relation Extraction”, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 241–251, Florence, Italy, 2019.

[8] Zhang Y., Qi P., Manning C.D., “Graph Convolution over Pruned Dependency Trees Improves Relation Extraction”, in: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2205–2215, Brussels, Belgium, 2018.

[9] Kipf T., Welling M., “Semi-Supervised Classification With Graph Convolution Networks”, in: *Conference paper at ICLR 2017*, April 24–26, 2017, Toulon, France.

[10] Veličkovic P. et al., “Graph Attention Networks”, in: *Conference paper at ICLR 2018*, April 39 – May 3, 2018, Vancouver, BC, Canada.

[11] Hamilton W.L., Ying R., Leskovec J., “Inductive representation learning on large graphs”, in: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 1025–1035, Curran Associates Inc., NY, USA, 2017.

[12] Bengio Y., Ducharme R., Vincent P., Jauvin C., “A Neural Probabilitic Language Model”, *Journal of Machine Learning Research*, Vol. 3, 1137–1155 (2003).

[13] Camacho-Collados J., Pilehvar M.T., “On the Role of Text Preprocessing in Neural Network Architectures: An Evaluation Study on Text Categorization and Sentiment Analysis”, in: *Proceedings of the 2018 EMNLP Workshop Blackbox NLP:*

Analyzing and Interpreting Neural Networks for NLP, 40–46, Brussels, Belgium, 2018.

[14] Soares L.B., FitzGerald N., Ling J., Kwiatkowski T., “Matching the Blanks: Distributional Similarity for Relation Learning”, in: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2895–2905, Association for Computational Linguistics, Florence, Italy, 2019.

[15] Mikolov T., et al., “Distributed Representations of Words and Phrases and their Compositionality”, in: *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Vol. 2, 3111–3119, Curran Associates Inc., NY, USA, 2013.

[16] Mikolov T., Chen K., Corrado G., Dean J., “Efficient Estimation of Word Representations in Vector Space”, in: *Proceedings of Workshop at ICLR*, 2013.

[17] Brown T.B., et al., “Language Models are Few-Shot Learners”, arXiv preprint arXiv:2005.14165. 2020.

[18] Devlin J., Chang M.-W., Lee K., Toutanova K., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, in: *Proceedings of NAACL-HLT 2019*, Minneapolis, Minnesota, June 2 – June 7, 2019, 4171–4186, Association for Computational Linguistics, 2019.

[19] Radford A., et al., “Language Models are Unsupervised Multitask Learners”, https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

[20] Peters M.E., et al., “Deep contextualized word representations”, in: *Conference paper at ICLR 2018.*

[21] Řehůřek R., 2009. “Gensim: Topic modelling for humans”, RARE Technologies Ltd., www.radimrehurek.com/gensim/

[22] Vaswani A., et al., “Attention Is All You Need”, in: *Proceedings of NIPS* 2017, Long Beach, CA, USA, 2017.

[23] Bahdanau D., Cho K., Bengio Y., “Neural Machine Translation by Jointly Learning to Align and Translate”, in: *Conference paper at ICLR 2015*.

[24] SpaCy. “Industrail-Strength Natural Language Processing”, www.spacy.io

[25] S. Hochreiter, J. Schmidthuber, “Long-Short Term Memory”, *Neural Computation*, Vol. Issue 8, 9 1735–1780 (1997).

[26] Zelenko D., Aone C., Richardella A., “Kernel Methods for Relation Extraction”, *Journal of Machine Learning Research*, Vol. 3, 1083–1106 (2003).

[27] Collins M., Duffy N., “Convolution Kernels for Natural Language”, *Advances in Neural Information Processing Systems 14, Proceedings of the 2001 Neural Information Processing Systems Conference* (*NIPS 2001*), 625–632, MIT Press, Cambridge, MA, USA, 2002.

[28] Cortes C., Vapnik V., “Support-vector Networks”, *Machine Learning*, 20, 273–297 (1995).

Przegląd metod reprezentacji tekstu w kontekście wyznaczania relacji semantycznych

M. GAŁUSZA

Artykuł zawiera przegląd najpopularniejszych metod reprezentacji tekstu – modele sekwencyjne i grafowe w kontekście wykrywania relacji semantycznych między słowami.

Słowa kluczowe: przetwarzanie tekstu, związki semantyczne, modele sekwencyjne, modele grafowe