A Mixed Neural Network and Support Vector Machine Model for Tender Creation in the European Union TED Database

Sangramsing Kayte[a] and Peter Schneider-Kamp[b]
Department of Mathematics and Computer Science (IMADA), University of Southern Denmark (SDU), Odense M, 5230, Denmark

Keywords: Natural Language Processing, Natural Language Understanding, Named-entity Recognition, Logistic Regression, European Union, Tender Electronic Daily, Common Procurement Vocabulary.

Abstract: This research article proposes a new method of automatized text generation and subsequent classification of the European Union (EU) Tender Electronic Daily (TED) text documents into predefined technological categories of the dataset. The TED dataset provides information about the respective tenders includes features like Name of project, Title, Description, Types of contract, Common procurement vocabulary (CPV) code, and Additional CPV codes. The dataset is obtained from the SIMAP-Information system for the European public procurement website, which is comprised of tenders described in XML files. The dataset was pre-processed using tokenization, removal of stop words, removal of punctuation marks etc. We implemented a neural machine learning model based on Long Short-Term Memory (LSTM) nodes for text generation and subsequent code classification. Text generation means that given a single line or just two or three words of the title, the model generates the sequence of a whole sentence. After generating the title, the model predicts the main applicable CPV code for that title. The LSTM model reaches an accuracy of 97% for the text generation and 95% for code classification using Support Vector Machine(SVM). This experiment is a first step towards developing a system that based on TED data is able to auto-generate and code classify tender documents, easing the process of creating and disseminating tender information to TED and ultimately relevant vendors. The development and automation of this system will future vision and understand current undergoing projects and the deliveries by a SIMAP-Information system for European public procurement tenders organisation based on the tenders published by it.

1 INTRODUCTION

Natural language processing (NLP) is a wide area concerned with the application of computers to analyze, understand, and derive meaning from human language in a smart and useful way. NLP can organize and structure knowledge to perform tasks such as automatic summarization, translation, named entity recognition, relationship extraction, sentiment analysis, speech recognition, and topic segmentation (Manning et al., 2014). The unswerving increase of categories for tender documents has resulted in the need for methods that can automatically classify new documents within subcategories of the text. In this paper we attempt to make the categorization process automated using a machine learning approach, i.e., to learn from the European public procurement tender data in Tender Electronic Daily (TED) database. Given text describing the tender, the model predicts codes using the common procurement vocabulary (CPV) that are relevant for this text. We commence this paper with an overview of the European Union (EU) TED and then move to the introduction of a few terminologies which are relevant to this paper. TED works under the Policies of European Public procurement as drawn by the Directorate General for Internal Market, Industry, Entrepreneurship, etc. A tender is a document that a purchasing agent publishes to announce its request for certain goods or services. The whole electronic tendering system is similar to that of a traditional one, i.e. a selling agent submits a bid to the purchasing organisation against the tender enquiry, which evaluates the quoting vendors based on the technical specifications as well as financial terms and conditions.

Then the purchasing agent announces the successful bidder. We carried out the classification

[a] https://orcid.org/0000-0002-9156-3794
[b] https://orcid.org/0000-0003-4000-5570

Kayte, S. and Schneider-Kamp, P.
A Mixed Neural Network and Support Vector Machine Model for Tender Creation in the European Union TED Database.
DOI: 10.5220/0008362701390145
In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*, pages 139-145
ISBN: 978-989-758-382-7

on the SIMAP-Information system for European public procurement tenders available on its website (https://ted.europa.eu/TED/main/HomePage.do). eNotices is the online tool for preparing public procurement notices and publishing them in the Supplement to the Official Journal of the EU.

There is a standard format for the calling of tenders in public procurement in European nations, which are referred online with the process, the Tender European Daily (TED) database. In this, there are different sections allotted to various categories like construction, financing, operation, and distribution of water bottling facilities, etc. The syntactically-parsed information is maintained in these TED tenders (Gelderman et al., 2006). The TED tenders are called from various sectors associated with public development and information is stored in the form of text and numbers. The important features of the data are given in Table 1.

The primary goal of this research is to classify tenders according to (parts of) the title of the project. Titles are generated from the user-given text. The implemented system achieves an accuracy of 97% when generating titles from user-given text and accuracy of 95% when predicting CPV codes from the generated titles. After a deep inspection of the dataset, it becomes apparent that the CPV code describes the uniqueness of each tender. Here, we tried to extract information about certain tenders in TED based on year-wise allotment along with their CPV codes. So when the CPV code is given it retrieves the information about the project.

In Section 2 of this paper, work related to the proposed method is described. Section 3 reviews relevant natural processing techniques. Section 4 describes the corpus of tenders from TED used in this project. The proposed framework is detailed in Section 5. Section 6 covers a preliminary experimental evaluation of the framework. We conclude in Section 7.

2 RELATED WORK

The neurons interconnected with each other simulate the network structure of neurons in the human brain (Le et al., 2011). The nodes are interconnected by edges, and then nodes are organized into layers. Computation and processing are done in a feed-forward manner, and errors can be back-propagated to previous layers to adjust the weights of corresponding edges (Hoskins et al., 1990). The hidden nodes are randomly assigned to avoid the extra load pay-off in the structure. In a simple network, the weights are usually learned in one single step which results in faster performance (Towell and Shavlik, 1993). For more complex relations, deep learning methods with multiple hidden layers are adopted. These methods were made more feasible with the advances of computing power in hardware and the Graphics Processing Unit (GPU) (Cireşan et al., 2012).

This allows them to deal with a time sequence with temporal relations such as speech recognition (Waibel et al., 1995). According to a previous comparative study of RNNs in NLP, RNNs are found to be more effective in sentiment analysis (Wang et al., 2016). Thus, we focus on RNNs with Long Short-Term Memory (LSTM) in this paper. As the time sequence grows in LSTM, it's possible for weights to grow beyond control or to vanish. To deal with the vanishing gradient problem in training RNNs and LSTM has been proposed to learn long-term dependency among longer time period (Graves et al., 2005). In addition to input and output gates, forget gates are added in LSTM. They are often used for time series prediction and hand-writing recognition (Sundermeyer et al., 2012). The shortfall of LSTM is that they are only able to make use of the previous context. To avoid this, LSTM is designed for processing the data in both directions with two separate hidden layers, which are then feed forwards to the same output layer. Using LSTM will run your inputs in two ways, one from past to future and one from future to past. This will help to improve the results and understand the context in a better way (Wang et al., 2018).

For NLP, it is useful to analyze the distributional relations of word occurrences in documents (Joachims, 2002). The simplest way is to use one-hot encoding to represent the occurrence of each word in documents as a binary vector (Turian et al., 2010). In distributional semantics, word embedding models are used to map from the one-hot vector space to a continuous vector space in a much lower dimension than conventional bag-of-words (BoW) model (Cho et al., 2014). Among various word embedding models, the most popular ones are distributed representation of words such as Word2Vec and GloVe, where neural networks are used to train the occurrence relations between words and documents in the contexts of training data (Pennington et al., 2014). In this paper, the Word2Vec word embedding model is used to represent words in short texts. Then, LSTM classifiers are trained to capture the long-term dependency among words in short texts. The sentiment of each text can then be classified as positive or negative (Wang et al., 2016). In this paper, LSTM is utilized in learning sentiment classifiers of short texts, like title and CPV code.

Table 1: Project features of data.

Features	Data type	Values
Name	Text	singe line free text
Description	Text	multiple line free text
Types of contract	Text	one of WORKS, SUPPLIES, or SERVICES
CPV code	Number	from predefined hierarchical catalog of codes
additional CPV codes	List of numbers	from predefined hierarchical catalog of codes

3 NATURAL LANGUAGE PROCESSING

NLP investigates the use of computers to process or to understand human languages for the purpose of performing useful tasks like information filtering, language identification, readability assessment, and sentiment analysis, etc. NLP is an interdisciplinary field that is a combination of the subfield of computer science, information engineering, and artificial intelligence. From a scientific perspective, NLP aims to model the cognitive mechanisms underlying the understanding and production of human languages (Deng and Liu, 2018). The following are the important pipeline of NLP talks processing:

Text Segmentation.
In this method, the sentence is broken into a sequence of contiguous parts called segments. There exist another sequence segmentation like sentence segmentation breaking a piece of string into sentences which is an important post-processing stage for speech transcription and chunking also known as shallow parsing to find important phrases from sentences, such as noun phrases (Pak and Teh, 2018).

Named Entity Recognition (NER).
NER is the process of finding and tagging named entities existing in the given text into pre-defined categories. The NER task is hugely dependent on the knowledge base used to train the NER extraction algorithm, so it may or may not work depending upon the provided dataset it was trained on.

Word Embedding.
A word embedding learning method is used to calculate these embeddings using distributional semantics; that is, they approximate the meaning of a word by considering how often it co-occurs with other words in the corpus. These embedding-based metrics usually approximate sentence-level embeddings using some heuristic to combine the vectors of the individual words in the sentence. The sentence-level embedding between the generated and reference response are compared using a measure such as a cosine distance (Mikolov et al., 2013).

Word2vec.
Word2vec is a two-layer neural net used in the recurrent neural network that processes text. Its input is a text corpus and its output is a set of vectors. It gives the best results for any kind of sequence neural network models to learn node and sequence representations based on node sequences (Liang et al., 2016).

Bag-of-Words (BOW).
The main idea of BOW is to predict the target word with surrounding context words. For convenient, the surrounding words are symmetric (so as in skip-gram), i.e., a window with size m is predefined and the task is to predict the target word w_c with a sequence of words $(wc-m,...,wc-1,wc+1,...,wc+m)$, where w_i denotes the word at position c (Le and Mikolov, 2014).

Text Cleaning.
The text cleaning is needed for converting the text into a particular input format. This helps to remove the tagged data, punctuation, stop words, abbreviations from the given input text (Hardeniya et al., 2016). Following are a few steps to clean the data:

- Removing the HTML tags
- Removing Punctuation and Stopwords
- Stemming
- Conversion of data text to lower case

4 CORPUS DESCRIPTION

The corpus consists of the information about the announcements and calls related to tenders in TED. The data is stored in an open XML format. We extracted it to a Comma-Separated Values (CSV) format, containing the information in form of text and numbers as indicated in Table 1. The CSV file contains information like the title of the project and CPV code as an important entity along with other sub-entities like description, reference of the project. The other entities like types of contract and a detailed description of a particular project are also part of the dataset. In this way, we are having a dataset of approx. 40,000 texts for each year from 2015 to 2019. This data will be helpful for training and implementing the machine learning model for the required system.

5 PROPOSED FRAMEWORK

The proposed framework consists of NLP library for pre-processing such using Natural Language Toolkit (NLTK) library and unidirectional LSTM architecture for sequence prediction and classification. We have also applied linear support vector machine (SVM) classifier to classify CPV main code category from other relevant fields, which is shown in the architecture in Figure 1.

The tender category of the title describe overall information about the particular product. Initially, first 4-6 words from the title are given as input for the system, the LSTM model generates the overall text information. At the first stage, pre-processing and word features are extracted. Secondly, the Word2Vec word embedding model is used to learn word representations as vectors.

The external input gate unit $g_i^{(t)}$ is computed similarly to the forget gate (with a sigmoid unit to obtain a getting value between 0 and 1), but with its own parameters of weights and model: (Hochreiter and Schmidhuber, 1997)

$$g_i(t) = \sigma\left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)}\right) \quad (1)$$

The output of the LSTM cell can also be shut off, via the output gate $q_i^{(t)}$, which also uses a sigmoid unit for gating: (Hochreiter and Schmidhuber, 1997)

$$q_i(t) = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)}\right) \quad (2)$$

which has parameters b^o, U^o, W^o for its biases, input weights and recurrent weights, respectively. Among the variants, one can choose to use the cell state $q_i^{(t)}$ as an extra input (with its weight) into the three gates of the $i-th$ unit, as shown in the above equation. Thus the LSTM model takes the recurrent input of weights and concurrently gives the output to the cell state (Hochreiter and Schmidhuber, 1997).

5.1 Preprocessing

In this stage, we filtered the text data into required format needed for the training of the system. The preprocessing is divided into four different steps:

1. **Cleaning** consists of getting rid of the less useful parts of a text through stopword removal, dealing with capitalization and characters, and other minor details.
2. **Annotation** This process involves the application of a scheme to texts which include structural markup and part-of-speech tagging.
3. **Normalization** consists of the translation (mapping) of terms in the scheme or linguistic reductions through Stemming, Lemmatization, and other forms of standardization.
4. **Analysis** consists of statistically probing, manipulating, and generalizing from the dataset for feature analysis.

5.2 Tokenization

Tokenization is the process of splitting text fields into meaningful segments by locating the word boundaries like CPV code, title from the given text database. The points where one word ends and another begins. For computational linguistic purposes, the words thus identified are frequently referred to as tokens. In written languages where no word boundaries are explicitly marked in the writing system, tokenization is also known as word segmentation, and this term is frequently used synonymously with tokenization.

5.3 Word Embedding

Word embeddings is the representation of title, description of words in the form of vectors. The aim is to represent words via vectors such that similar text words or words used in a similar context are close to each other while antonyms end up far apart in the vector space.

The conversion of word vectors into a high-dimensional space can be envisioned as shown in Figure 2

5.4 Dense Layer

The dense layer contains a linear operation in which every input is connected to every output by weight (so there are n_inputs * n_outputs weights - which can be a lot!). Generally, this linear operation is followed by a non-linear activation function.

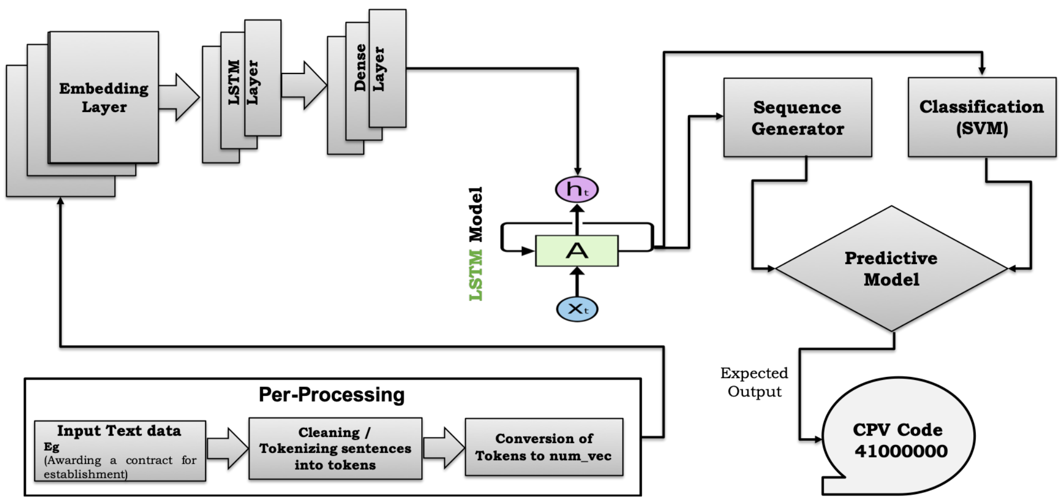

Figure 1: The system architecture of the proposed approach.

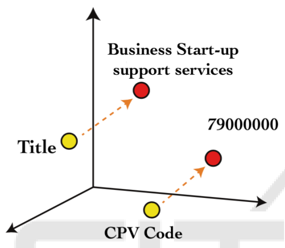

Figure 2: Visualization of Word Embeddings.

5.5 Support Vector Machines

SVM is text classification approach which can be applied in both types of classification and regression problems. The feature of SVM model find a maximum marginal hyperplane(MMH) that best divides the dataset into classes and minimization of error. The classifier identifies the unique data points using a hyperplane with the largest amount of margin using training data.SVM finds an optimal hyperplane which helps in classifying new data points and so also referred as discriminative classifier (Sassano, 2003).

The decision function g in SVM framework is defined as: (Sassano, 2003)

$$g(x) = sgn(f(x)) \tag{3}$$

$$f(x) = \sum_{i=1}^{l} y_i \alpha K(x_i, x) + b \tag{4}$$

where K is a kernel function and σi are weights

5.6 Long Short-Term Memory (LSTM)

RNNs have a powerful sequence study capability, which can finely describe the dependence relationship of title and CPV code data (Zhang and Lu, 2018). RNNs such as LSTM are powerful models that are capable of learning effective feature representations of sequences when given enough training data (Zhang et al., 2016). RNNs are layered neural networks that include recurring connections between layers. The recurrence creates a temporal effect on the network, allowing certain network connections (parameters) to be used at different times. This allows RNNs to capture temporal relationships from a data sequence, which make them appropriate for predicting sequential data.

The LSTM deals especially in handling sequences that have a large gap between relevant information for learning or predictions of new features in the training set. The attractive feature of LSTM is that it effectively deals with the long-term dependence problem in time sequences (Sutskever et al., 2014).

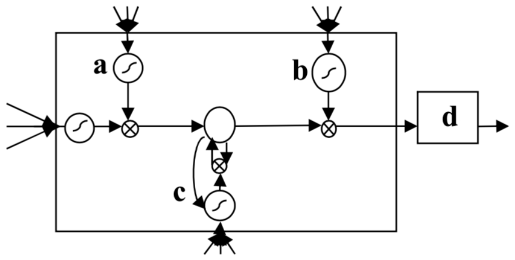

Figure 3: The LSTM cell. a-Input gate, c-Forget gate, b-Output gate, d- output from the cell (Skovajsová, 2017).

Figure 3 shows the structure of the LSTM network, where the repeated modules represent the hidden layer in each iteration. Each hidden layer contains a number of neurons. Each neuron performs a linear matrix calculation on the input vector and then outputs the corresponding results after the nonlinear action of the activation function (Arjovsky et al., 2016).

In each iteration, the output of the previous iteration interacts with the next word vector of the text determines the preservation or abandonment of the information and the update of the current state. h_t is the input of the hidden layer in this iteration. According to the current state information, the predicted output value of hidden layer y is obtained, and the output vector h_t is provided for the next hidden layer at the same time. Whenever there is a new word vector input in the network, the output of the hidden layer at the next moment is calculated in conjunction with the output of the hidden layer at the last moment. The hidden layer circulates and keeps the latest state (Poliak et al., 2017)

6 EXPERIMENTAL EVALUATION

We have calculated the accuracy of our NLP model using text prediction and classification. This involves the concepts of true positives, true negatives, false positives, and false negatives are also evaluated. False positives are cases the model incorrectly labels as positive that are actually negative, or, in our example, the text which is not identified is considered a false negative. True positives are data points classified as positive by the model that actually are positive (meaning they are correct), and false negatives are data points the model identifies as negative that actually are positive (incorrect). The LSTM-based model we implemented achieves an accuracy of 97% for the text generation and 95% for the code classification.

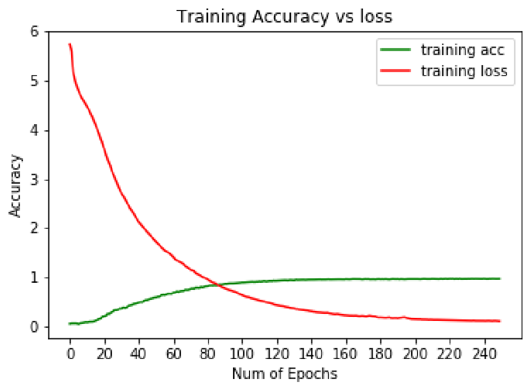

Figure 4: ROC curve against the training accuracy and loss.

An Receiver operating characteristic (ROC) curve plots the accuracy on the x-axis against the number of epochs on y-axis. The true positive rate (TPR) is the recall and the false positive rate (FPR) is the probability of a false alarm. Figure 4 shows the ROC curve of training accuracy and training loss of the system against the number of epochs. We can read from observations that the training accuracy gradually increases and loss decreases as the number of epochs for training of the system progresses. This generates and classifies the data accurately from the given dataset of the model.

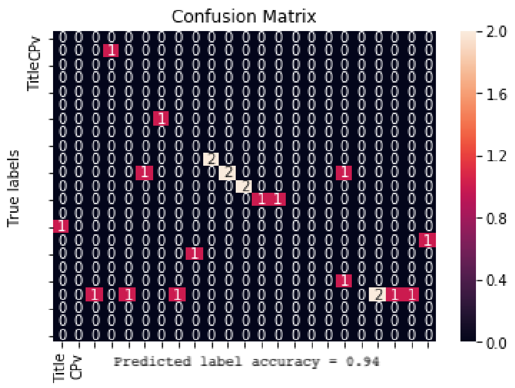

Figure 5: Confusion Matrix training accuracy.

Figure 5 confusion matrix is a summary of prediction results on a classification of text data from given dataset. In this matrix, the number of correct and incorrect predictions like title of project and CPV code are summarized with count values and broken down by each class. The confusion matrix helps to identify the ways in which your classification model is confused when it makes predictions. It also clears the errors being made by your classifier but more importantly the types of errors that are being made. So in this project, the confusion matrix is applied for the classification of text and the percentage accuracy is 95% where the title and CPV code is exactly identified from the given text data.

7 CONCLUSIONS

This experimental paper reported a high-accuracy approach based on an LSTM model for predicting CPV codes for TED tenders using a two step approach of generating texts and subsequently classifying the generated texts. Future work is to further increase the performance of the system as well as to automate the remaining steps of TED tender creation to the fullest extent possible.

REFERENCES

Arjovsky, M., Shah, A., and Bengio, Y. (2016). Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, pages 1120–1128.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cireşan, D., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classification. *arXiv preprint arXiv:1202.2745*.

Deng, L. and Liu, Y. (2018). *Deep Learning in Natural Language Processing*. Springer.

Gelderman, C. J., Ghijsen, P. W. T., and Brugman, M. J. (2006). Public procurement and EU tendering directives–explaining non-compliance. *International Journal of Public Sector Management*, 19(7):702–714.

Graves, A., Fernandez, S., and Schmidhuber, J. (2005). Bidirectional LSTM networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*, pages 799–804. Springer.

Hardeniya, N., Perkins, J., Chopra, D., Joshi, N., and Mathur, I. (2016). *Natural Language Processing: Python and NLTK*. Packt Publishing Ltd.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hoskins, J. C., Kaliyur, K. M., and Himmelblau, D. M. (1990). Incipient fault detection and diagnosis using artificial neural networks. In *IJCNN 1990, International Joint Conference on Neural Networks, San Diego, CA, USA, June 17-21, 1990*, pages 81–86.

Joachims, T. (2002). *Learning to classify text using support vector machines*, volume 668. Springer Science & Business Media.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196.

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2011). Building high-level features using large scale unsupervised learning. *arXiv preprint arXiv:1112.6209*.

Liang, D., Altosaar, J., Charlin, L., and Blei, D. M. (2016). Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Proceedings of the 10th ACM conference on recommender systems*, pages 59–66. ACM.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Pak, I. and Teh, P. L. (2018). Text segmentation techniques: a critical review. In *Innovative Computing, Optimization and Its Applications*, pages 167–181. Springer.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Poliak, A., Rastogi, P., Martin, M. P., and Van Durme, B. (2017). Efficient, compositional, order-sensitive n-gram embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 503–508.

Sassano, M. (2003). Virtual examples for text classification with support vector machines. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 208–215. Association for Computational Linguistics.

Skovajsová, L. (2017). Long short-term memory description and its application in text processing. In *2017 Communication and Information Technologies (KIT)*, pages 1–4. IEEE.

Sundermeyer, M., Schluter, R., and Ney, H. (2012). LSTM neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, pages 194–197.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Towell, G. G. and Shavlik, J. W. (1993). Extracting refined rules from knowledge-based neural networks. *Machine learning*, 13(1):71–101.

Turian, J., Ratinov, L., and Bengio, Y. (2010). Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. J. (1995). Phoneme recognition using time-delay neural networks. *Backpropagation: Theory, Architectures and Applications*, pages 35–61.

Wang, J., Yu, L.-C., Lai, K. R., and Zhang, X. (2016). Dimensional sentiment analysis using a regional CNN-LSTM model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 225–230.

Wang, J.-H., Liu, T.-W., Luo, X., and Wang, L. (2018). An LSTM approach to short text sentiment classification with word embeddings. In *Proceedings of the 30th Conference on Computational Linguistics and Speech Processing (ROCLING 2018)*, pages 214–223.

Zhang, K., Xu, J., Min, M. R., Jiang, G., Pelechrinis, K., and Zhang, H. (2016). Automated it system failure prediction: A deep learning approach. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 1291–1300. IEEE.

Zhang, Y. and Lu, X. (2018). A speech recognition acoustic model based on lstm-ctc. In *2018 IEEE 18th International Conference on Communication Technology (ICCT)*, pages 1052–1055. IEEE.