A Neural NLP Framework for an Optimized UI for Creating Tenders in the TED Database of the EU

Sangramsing N Kayte

Department of Mathematics and
Computer Science (IMADA),
University of Southern Denmark (SDU),
Odense, Denmark
Email: sangram@imada.sdu.dk

Peter Schneider-Kamp

Department of Mathematics and
Computer Science (IMADA),
University of Southern Denmark (SDU),
Odense, Denmark
Email: petersk@imada.sdu.dk

***Abstract*—The developments in the fields of web technologies, digital libraries, technical documentation, and medical data have made it easier to access a larger amount of textual documents, which can be combined to develop useful data resources. Textmining or the knowledge discovery from textual databases is a challenging task, in particular when having to meet the standards of the depth of natural language that is employed by most of the available documents. The primary goal of this research is to implement the Machine Learning algorithms like RecurrentNeural Networks (RNN) and Long Short Term Memory networks(LSTM) techniques that is applied for text mining / text prediction in the context of creating an optimized user interface for tender creation in the Tender Electronic Daily database used for public procurement throughout the European Union (EU). This paper explains the scope and concept of Natural Language Processing(NLP) for such text mining projects. We also present a detailed overview of the different techniques involved in Natural language processing and Understanding along with the related work.**

Keywords–Natural Language Processing, Natural Language Understanding, Natural User Interfaces, Named-Entity Recognition, Logistic Regression, European Union, Tender Electronic Daily, Common Procurement Vocabulary

I. INTRODUCTION

Natural language processing (NLP) is a emerging field which involves the combination of different fields like linguistics, computer science, and artificial intelligence leveraged to build assertive devices for variants of language assistance [1]. The foundation of NLP is laid down with the version of punch cards and now evolved to the heights of computing operations with infractions. [2]. NLP is broaden area to perform a wide range of natural language-related tasks at all levels like parsing, Part-Of-Speech (POS) tagging to different forms of machine response [3].

The advancement of Machine learning algorithms and deep learning architectures in NLP research areas is demanding and increasing every day. The early experiments in machine learning for NLP was targeting NLP problems which were based on shallow models (e.g., support vector machines and logistic regression) trained on very high dimensional and sparse features. But last few years, the neural networks based on dense vector representations producing outstanding superior results on various NLP tasks [4]. The significant improvement is with the implementation of word embeddings and deep learning methods. Deep learning enables multi-level automatic feature representation learning. This has replaced the traditional machine learning techniques used in NLP systems which relies heavily on hand-crafted features, most of them are time-consuming and often incomplete [5].

The NLP task like named-entity recognition (NER), semantic role labelling (SRL), and POS tagging are the methods proving the outstanding performance to solve difficult NLP tasks [6]. We reviewed different deep learning-related models and methods applied to natural language tasks such as a convolutional neural network (CNN, or ConvNet), Recurrent Neural Networks (RNN), and recursive neural networks. We also presented memory-augmenting strategies, attention mechanisms, and unsupervised models, reinforcement learning methods applied for language-related tasks [7].

In public procurement in the European Union (EU), tenders are called online via the Tender European Daily (TED) database. Tenders in TED are called under different categories like parking allotment, factories, etc. This dataset consists of syntactically-parsed information about different types of TED tenders [8]. The TED tenders are called from different fields and information is stored in the form of text and numbers. The important features of the data are given in the following table.

The primary goal of this research is to develop an optimizing user interface for tender creation in the context of TED using text mining techniques and applying Machine Learning (ML) algorithms for automation of the entire TED process. After a deep study of the dataset, it is understood that the CPV code describes the uniqueness of each type of 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 II of this paper, work related to the proposed method is described. Section III reviews relevant natural processing techniques. The proposed framework and corpus description is detailed in Section IV. Section V covers a preliminary experimental evaluation of the framework. We conclude in Section VI.

II. RELATED WORK

The neurons interconnected with each other simulate the network structure of neurons in the human brain [9]. The nodes are interconnected by edges, and then nodes are organized into layers. Computation and processing are done in a feed-forward

 ISBN: 978-1-61208-739-9

TABLE I. FEATURES OF DATA

Features	Data type	Values
Name of Project	Text	singe line free text
Description of Project	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

manner, and errors can be back-propagated to previous layers to adjust the weights of corresponding edges [10], [11]. 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 [12]. For more complex relations, deep learning methods with multiple hidden layers are adopted. These methods were made feasible thanks to the recent advances of computing power in hardware, and the Graphics Processing Unit (GPU) processing in software technologies [13]. Depending on the different ways of structuring multiple layers, several types of deep neural networks were proposed, where ConvNet and RNN are among the most popular ones [14]. ConvNet is usually used in computer vision since convolution operations can naturally be applied in edge detection and image sharpening [15]. They are also useful in calculating weighted moving averages and calculating impulse responses from signals [16]. RNNs are a type of neural networks where the inputs of hidden layers in the current point of time depend on the previous outputs of the hidden layer [17].

This allows them to deal with a time sequence with temporal relations such as speech recognition [18]. According to a previous comparative study of RNN and ConvNet in NLP, RNN is found to be more effective in sentiment analysis than ConvNet [19]. Thus, we focus on RNN in this paper. As the time sequence grows in RNNs, it's possible for weights to grow beyond control or to vanish. To deal with the vanishing gradient problem in training conventional RNN, bidirectional Long Short-Term Memory (BLSTM) has been proposed to learn long-term dependency among longer time period [20]. In addition to input and output gates, forget gates are added in BLSTM [21]. They are often used for time-series prediction and hand-writing recognition. The shortfall of conventional BLSTM is that they are only able to make use of the previous context. To avoid this, BLSTM 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 BLSTM 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.
For NLP, it is useful to analyze the distributional relations of word occurrences in documents [22]. The simplest way is to use one-hot encoding to represent the occurrence of each word in documents as a binary vector [23]. 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 [24]. 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 [25]. In this paper, we adopt the Word2Vec word embedding model to represent words in short texts. Then, BLSTM 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 [19]. In this paper, we utilize BLSTM in learning sentiment classifiers of short texts [26].

III. NATURAL LANGUAGE PROCESSING

The most important application of NLP is to make machines understand, respond to, and communicate with human languages. Thus, it acts as a bridge to improve the performance of communication [27]. With the growth and extension of machine learning and deep learning modules in different fields along with NLP, it increases the performance and efficiency of the system [28]. It is divided into the following categories.

a) Parsing:
Parsing is a technique of breaking up sentences according to grammar rules. A sentence is broken into a Noun Phrase and Verb Phrase. The Noun Phrase could again be divided into Article and Noun. This helps to convert the text into required grammar formats [29].

b) Stemming:
Stemming is the process of reducing inflexion in words to their root forms, e.g. by mapping a group of words to the same stem even if the stem itself is not a valid word in the language [30]. The stemming a word or sentence may result in words that are not the actual word. In stemming the 'stem' is obtained after applying a set of rules but without bothering about the part-of-speech (POS) or the context of the word occurrence [31].

c) Text Segmentation:
Text Segmentation is the division of sentences into smaller parts called segments. These segments can be words, sentences, topics, phrases, or any information unit depending on the task of the text analysis [32].

d) Named Entity Recognition (NER):
Named entity recognition is the process of allocating names to different entities like a person, location, organization, drug, time, clinical procedure, or biological protein in text. NER systems are often used as the first step in question answering, information retrieval, co-reference resolution, topic modelling, etc, [33].

e) Sentiment Analysis:
Sentiment analysis involves the aspects covered in the text with respect to the writer's perspective. The focus is on identifying some topics or the overall sentiment polarity of a text, such as positive or negative [34].

f) Word Embedding:
Word embedding is a big part of NLP-related research works. A word embedding with word2vec determines the syntactic properties of words based on co-occurrence in a text corpus and

 ISBN: 978-1-61208-739-9

assigns to each of the words a vector in the vector embedding space. Word embedding of sentences can determine the word characteristics and context [35].

1) **Sentiment Analysis**:- The word embedding has been proven to improve the performance of sentiment classification systems [36].
2) **Sentiment-Specific**:- The generic word embedding only considers semantic relations from the words; it fails to differentiate sentiment words such as "good" or "bad".

g) Word2vec:

Word2vec is an embedding technique which is the evaluation of a set of word vectors [37]. It is also known as distributed word representation that can capture both the semantic and syntactic information of words from a large unlabelled corpus. Words that occur in similar contexts tend to have similar meanings and are labelled accordingly with the word2vec method [38].

1) **Continuous Bag-of-Words** This is the first proposed architecture which is similar to the feed-forward neural network language models (NNLM), where the non-linear hidden layer is removed and the projection layer is shared for all words (not just the projection matrix); thus, all words get projected into the same position (their vectors are averaged). The advantage of this method is that as more gets added in future, it does not effect the order of words in history. The training complexity is then:

$$Q = ND + Dlog2(V) \tag{1}$$

We denote this model further as CBOW, as unlike the standard bag-of-words model, it uses a continuous distributed representation of the context [39].
2) **Skip-gram Model** The Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document [40]. More formally, given a sequence of training words $w_1, w_2, w_3, \ldots, w_T$, the objective of the Skip-gram model is to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_t + j|w_t) \tag{2}$$

where c is the size of the training context, which can be a function of the centre word wt. Larger c results in more training examples and, thus, can lead to higher accuracy at the expense of the training time [41].

h) Bag-of-Words:

The Bag-of-Words (BoW) model learns a vocabulary from all of the documents and then models each document by counting the number of times each word appears. The BoW model is a simplifying representation used in NLP and information retrieval (IR). In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity [42].

i) Stop Words:

Text may contain stop words like 'the', 'is', 'are'. Stop words can be filtered from the text to be processed. There is no universal list of stop words in NLP research, however, the NLTK module contains a list of stop words [18]. Consequently, such types of words can be processed separately.

IV. Proposed Framework

The proposed framework consists of preprocessing, feature extraction, word embedding, and BLSTM classification. The corpus description for the overall architecture is related to the information about the announcement and calls related to tenders in TED. The data is stored in an open XML format. We extracted it to CSV format, containing the information in form of text and numbers as indicated in Table I. Recall that the CSV file contains information like the title of the project and CPV code as an important entity. The other entities like types of contract and detailed description about a particular project are also part of the data set. In this way, we are having a dataset of approx, $40{,}000$ texts for each year from 2015 to 2018. The architecture flow is shown in Figure 1.

As shown in the Figure 1, short texts are first pre-processed and word features are extracted. Second, the Word2Vec word embedding model is used to learn word representations as vectors.

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

$$q_i(t) = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)}\right) \tag{3}$$

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 above equation. Thus the BLSTM model takes the recurrent input of weights and concurrently gives the output to the cell state.

A. Preprocessing

None of the magic described above happens without a lot of work on the back end. Transforming text into something an algorithm can digest is a complicated process. 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** consists of the application of a scheme to texts. Annotations 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.

B. Tokenization

Tokenization is the process of breaking up the sequence of characters in a text by locating the word boundaries, 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.

 ISBN: 978-1-61208-739-9

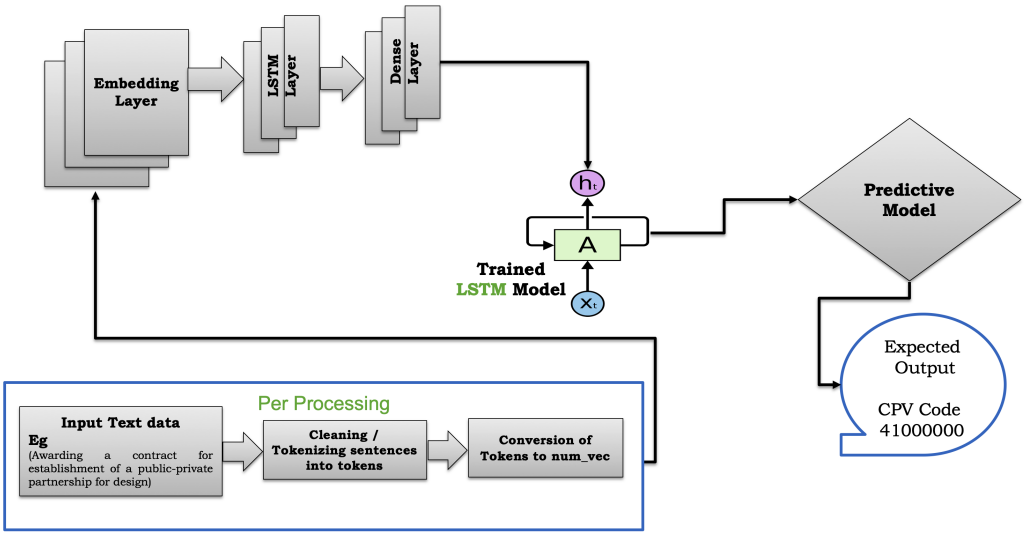

Figure 1. The system architecture of the proposed approach

C. Word Embedding

Simply put, word embeddings let us represent words in the form of vectors. But these are not random vectors, the aim is to represent words via vectors such that similar words or words used in a similar context are close to each other while antonyms end up far apart in the vector space.

D. 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.

E. Long Short-Term Memory (LSTM)

The architecture of LSTM is a combination of recurrent neural networks and feed-forward neural network [43]. LSTM has numerous applications related to text document processing based on context solving for the given task. So it predicts the next coming term from the relative context thus, words are not treated as independent individuals, but as the units dependent on their immediate neighbourhood in the text. LSTM advantage it is that the output does not depend on the length of input because the input is entered sequentially, one input per time step [44]. The basic architecture of the LSTM recurrent neural network receipts the memory block, that consists of several memory cells with which one can communicate by the input gate, forget gate, and output gate of that cell. The training of the LSTM neural network is performed in two phases, forward pass and backward pass. The important feature to note is that LSTM memory cells give different roles to addition and multiplication in the transformation of inputs. The central plus sign in both diagrams below is essentially the secret of LSTM. Figure 2 shows the architecture of a simple recurrent network with one input and one output gate. The working architecture of LSTM is clearly described on the right-hand side of Figure 2. Instead of determining the subsequent cell state by multiplying its current state with new input, they add the two, and that quite literally makes the difference. The forget gate still relies

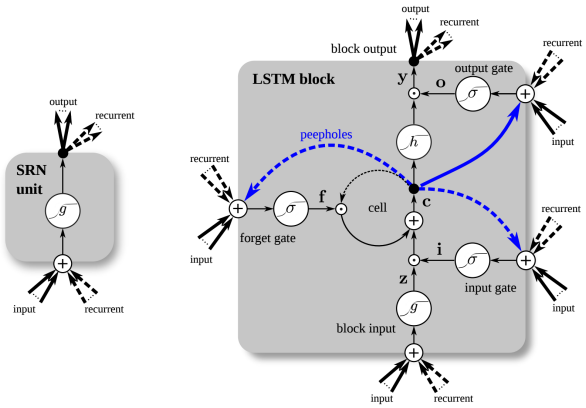

Figure 2. The Simple Recurrent Network unit (left) and a LSTM block (right) as used in the hidden layers of a recurrent neural network [45].

on multiplication, of course. Different sets of weights filter the input for input, output, and forgetting. The forget gate is represented as a linear identity function. The reasons for this is that if the gate is open, the current state of the memory cell is simply multiplied by one, to propagate forward one more time step [41].

V. Baseline Results from Experimental Evaluation

We have calculated the accuracy of our NLP model using Precision and Recall. These ways of evaluation use the concepts of true positives, true negatives, false positives, and false negatives. Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives. 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. Our trained model achieves a precision of 95.5% where it accurately identifies the sequence prediction for the given initial text words and identifies the respective CPV code. This gives a very good accuracy as it resolves ambiguity in predicting the sequence.

 ISBN: 978-1-61208-739-9

The research study benchmarks shows the Defence Research and Development Organisation (DRDO) lab (India) which prediction and classification technique applied to the vendor's system where it predicts and occurrences of the database [46].

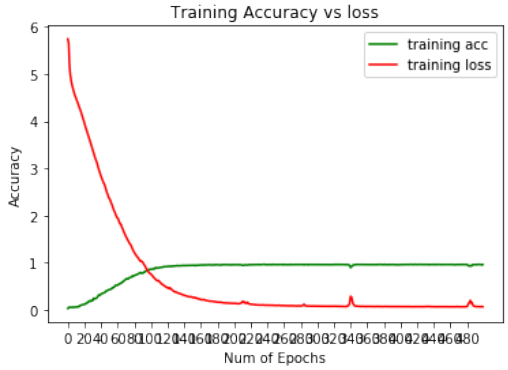

Figure 3. ROC curve against the training accuracy and loss

The recall is the number of true positives divided by the number of true positives plus the number of false negatives. True positives are data points classified as positive by the model that actually is positive (meaning they are correct), and false negatives are data points the model identifies as negative that actually are positive (incorrect). A ROC curve plots the true positive rate on the y-axis versus the false positive rate on the x-axis. The true positive rate (TPR) is the recall and the false positive rate (FPR) is the probability of a false alarm. Figure 3 shows the ROC curve of training accuracy and training loss of the system. It is observed that after the increase in a number of epochs the training accuracy increases and loss becomes constant. This makes our system more stable with improvements in performance accuracy.

VI. Conclusion

This project work focuses on the automation and optimization of tender creation in the European TED system. TED data sequence generation is a complex task. In most TED titles there is 25 %-30 % sequence of text is similar that makes more ambiguity to predict the correct sequence. In our case, the models with 85 % couldn't resolve complete ambiguity we need model accuracy more than 90 %. We have also presented the literature review with respect to NLP and text mining concepts. The implement of this algorithm is the initial but an important step towards an optimized interface for tender creation. The final goal is a natural user interface, improving user experience by automation and increasing human efficiency significantly. In future work, we can develop a GUI-based model with the proposed algorithm for making the optimized process for calling TED.

References

[1] J. Hirschberg and C. D. Manning, "Advances in natural language processing," Science, vol. 349, no. 6245, 2015, pp. 261–266.

[2] K. S. Jones, "Natural language processing: a historical review," in Current issues in computational linguistics: in honour of Don Walker. Springer, 1994, pp. 3–16.

[3] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [review article]," IEEE Comp. Int. Mag., vol. 13, no. 3, 2018, pp. 55–75.

[4] Y. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," Nature, vol. 521, no. 7553, 2015, pp. 436–444.

[5] S. Marcel, M. S. Nixon, J. Fierrez, and N. W. D. Evans, Eds., Handbook of Biometric Anti-Spoofing - Presentation Attack Detection, Second Edition, ser. Advances in Computer Vision and Pattern Recognition. Springer, 2019.

[6] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in Proceedings of the 25^{th} International conference on Machine learning. ACM, 2008, pp. 160–167.

[7] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for natural language processing," arXiv preprint arXiv:1606.01781, vol. 2, 2016.

[8] C. J. Gelderman, P. W. T. Ghijsen, and M. J. Brugman, "Public procurement and EU tendering directives–explaining non-compliance," International Journal of Public Sector Management, vol. 19, no. 7, 2006, pp. 702–714.

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

[10] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. S. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images." in ICML, vol. 1, no. 2, 2016, pp. 1–4.

[11] J. C. Hoskins and D. Himmelblau, "Artificial neural network models of knowledge representation in chemical engineering," Computers & Chemical Engineering, vol. 12, no. 9-10, 1988, pp. 881–890.

[12] G. G. Towell and J. W. Shavlik, "Extracting refined rules from knowledge-based neural networks," Machine learning, vol. 13, no. 1, 1993, pp. 71–101.

[13] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," arXiv preprint arXiv:1202.2745, 2012.

[14] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "CNN-RNN: a unified framework for multi-label image classification," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2285–2294.

[15] G. Chartrand, P. M. Cheng, E. Vorontsov, M. Drozdzal, S. Turcotte, C. J. Pal, S. Kadoury, and A. Tang, "Deep learning: a primer for radiologists," Radiographics, vol. 37, no. 7, 2017, pp. 2113–2131.

[16] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigne, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds," Speech communication, vol. 27, no. 3-4, 1999, pp. 187–207.

[17] M. Sundermeyer, R. Schluter, and H. Ney, "LSTM neural networks for language modeling," in Thirteenth annual conference of the international speech communication association, 2012, pp. 194–197.

[18] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," Backpropagation: Theory, Architectures and Applications, 1995, pp. 35–61.

[19] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "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), vol. 2, 2016, pp. 225–230.

[20] A. Graves, S. Fernandez, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in International Conference on Artificial Neural Networks. Springer, 2005, pp. 799–804.

[21] J.-H. Wang, T.-W. Liu, X. Luo, and L. Wang, "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), 2018, pp. 214–223.

[22] T. Joachims, Learning to classify text using support vector machines. Springer Science & Business Media, 2002, vol. 668.

[23] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics, 2010, pp. 384–394.

[24] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using

 ISBN: 978-1-61208-739-9

rnn encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014.

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

[26] M. Dwarampudi and N. Reddy, "Effects of padding on LSTMs and CNNs," arXiv preprint arXiv:1903.07288, 2019.

[27] A. Vinciarelli, M. Pantic, D. Heylen, C. Pelachaud, I. Poggi, F. D'Errico, and M. Schroeder, "Bridging the gap between social animal and unsocial machine: A survey of social signal processing," IEEE Transactions on Affective Computing, vol. 3, no. 1, 2012, pp. 69–87.

[28] I. Arel, D. C. Rose, T. P. Karnowski et al., "Deep machine learning-a new frontier in artificial intelligence research," IEEE computational intelligence magazine, vol. 5, no. 4, 2010, pp. 13–18.

[29] J. Kimball, "Seven principles of surface structure parsing in natural language," Cognition, vol. 2, no. 1, 1973, pp. 15–47.

[30] M. F. Porter, "Snowball: A language for stemming algorithms," pp. 1–4, 2001.

[31] I. A. Al-Sughaiyer and I. A. Al-Kharashi, "Arabic morphological analysis techniques: A comprehensive survey," Journal of the American Society for Information Science and Technology, vol. 55, no. 3, 2004, pp. 189–213.

[32] I. Pak and P. L. Teh, "Text segmentation techniques: a critical review," in Innovative Computing, Optimization and Its Applications. Springer, 2018, pp. 167–181.

[33] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in Proceedings of the thirteenth conference on computational natural language learning. Association for Computational Linguistics, 2009, pp. 147–155.

[34] B. Pang, L. Lee et al., "Opinion mining and sentiment analysis," Foundations and Trends® in Information Retrieval, vol. 2, no. 1–2, 2008, pp. 1–135.

[35] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in Advances in neural information processing systems, 2014, pp. 2177–2185.

[36] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for twitter sentiment classification," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, 2014, pp. 1555–1565.

[37] W.-L. Chao, S. Changpinyo, B. Gong, and F. Sha, "An empirical study and analysis of generalized zero-shot learning for object recognition in the wild," in European Conference on Computer Vision. Springer, 2016, pp. 52–68.

[38] Z.-Y. Niu, D.-H. Ji, and C. L. Tan, "Semi-supervised feature clustering with application to word sense disambiguation."

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[40] P. Liu, X. Qiu, and X. Huang, "Learning context-sensitive word embeddings with neural tensor skip-gram model," in Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015, pp. 1284–1290.

[41] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in Advances in neural information processing systems, 2013, pp. 3111–3119.

[42] J. Gao, Y. He, X. Zhang, and Y. Xia, "Duplicate short text detection based on word2vec," in 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS). IEEE, 2017, pp. 33–37.

[43] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," Journal of machine learning research, vol. 3, 2002, pp. 115–143.

[44] L. Skovajsová, "Long short-term memory description and its application in text processing," in 2017 Communication and Information Technologies (KIT). IEEE, 2017, pp. 1–4.

[45] K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: a search space odyssey," IEEE transactions on neural networks and learning systems, vol. 28, no. 10, 2017, pp. 2222–2232.

[46] S. Goswami, P. Bhardwaj, and S. Kapoor, "Naïve bayes classification of drdo tender documents," in 2014 International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, 2014, pp. 593–597.

 ISBN: 978-1-61208-739-9