Grenze International Journal of Engineering and Technology, Jul 2019

Easy Verdict: Digital Assistant to Resolve Criminal Litigation

Reeta Koshy[1], Abhishek Padalkar[2], Neha Nikam[3] and Vinit Jain[4]
[1-4]Dept. of Computer Engineering. Sardar Patel Institute of Technology, Mumbai, India
Email: reeta_koshy@spit.ac.in, {abhishekpadalkar97, nikam2neha, vinitgjain}@gmail.com

***Abstract*—A possible solution to the deficit in the number of judges as compared to the number of cases pending is an automated system that gives an accurate verdict after considering every detail of an offence. The system designed intends to provide reliable assistance in conjunction with the judges to dispense justice. As the system is at an experimental level, only simple cases with a binary output are considered (Sections 185 and 140 from the Motor Vehicle Act, 1988). MaLSTM RNN, an extension of the Siamese network that avoids diminishing gradient problem found in traditional RNN, is used. The system developed has an accuracy of approximately 90 percent. The scope of the system is a comprehensive system that can administer a verdict to any criminal offence regardless of its nature, which maps an offence with all the applicable sections and gives an inclusive verdict.**

***Index Terms*— Semantic Analysis, Siamese Network, Automated Law, Digital Assistant for Verdict and LSTM.**

I. INTRODUCTION

As of 2018, Indian courts have 3.3 crore cases pending. To solve them, we have 16119 judges in the subordinate judiciary, 598 in high courts and a mere 26 in the Supreme Court. The population of India is a staggering 133.92 crores, as per 2017, the second highest in the world, after China. Our judiciary system is slow because the number of judges cannot support such a huge population and hence, cases keep piling up. Over 1 crore cases are traffic related cases, and these cases clog up the lower courts, taking away valuable court time from more serious offences. The ratio of the population to the number of judges is 1000000:19, approximately (One million citizens to 19 judges). We have proposed a solution to this grave problem by creating a system that will solve simple traffic cases using Machine Learning, so they don't take up court time. This will make the entire judiciary system faster and more efficient, which is the need of the hour in the country. We want to start with simpler court cases like traffic related cases and overtime, move to more complicated cases, when the system becomes stronger and more robust. The cases that we have considered fall under the Driving Under Influence and Accident Related categories that fall under the Motor Vehicle Act 1988. The accused will be given a punishment according to his/her crime, which will be retrieved from a mapping to the Motor Vehicles Act 1988. First time offenders will be differentiated from the repeated ones. During the development of the system we have made a few assumptions, they are as follows:
(i) The accused is guilty of the crime and does not wish to challenge the account of the incident given in the FIR.
(ii) The account of the police officer filing the FIR is purely factual without any bias.

Grenze ID: 01.GIJET.5.2.3_1

II. Related Work

Ref. [5], The approach used a labeled dataset which had pairs of sentences involving compositional knowledge (SICK) using a wide spectrum of algorithms such as k-Nearest Neighbors, Random Forest, SVM and model ensembles. Zhao, Zhu. Ref. [6], Lan in 2014 proposed a unique methodology where they used the latent semantic analysis to represent in vector-space. Ref. [7], The approach used by Bjerva which employed formal semantics and logical references in collaboration with features extracted from word2vec model designed by Mikolov(2013) which is the primary input to the model designed in this project. Ref. [8], Tai, Socher, and Manning (2015) propose Tree-LSTMs which hypothesizes the chain-structure of general LSTMs to tree-structured topologies. Every sentence is converted to a parse tree using a trained parser, and the Tree LSTM evaluates its hidden state for a particular tree node of the corresponding word and the hidden states of every child node. Ref. [9], Gimpel, He and Lin (2015) have proposed a model which uses a complicated Convolution Network, abbreviated as ConvNet which determines the similarity between two sentences variation which induces sentence proximity in semanticity by incorporating different contrasts over numerous convolutions at different scales, they have used substantial architectural engineering as it was the training dataset that was relatively vague with a large spectrum of topics, our approach has overcome this problem by selecting a domain of the law (MVA, 1988) and applied it to two sections yielding us a better solution. Ref. [10], The Siamese architecture was used for the face verification, an application developed by Chopra, Hadsell, and LeCun in 2005, which employs symmetric ConvNets where we have used LSTMs. Siamese neural networks been implemented in a number of prediction application using ConvNets but to the best of our knowledge, recurrent neural networks is a largely untapped domain.

III. Methodology

Our goal is to create an accurate judgement for cases related to accidents and driving under influence(DUI). Knowing that every case will have some sort of description about the incident, we focus on that part to solve it. Now we assume that all cases are filed on behalf of different police officers, thus words used in every description about the cases are different. But semantically all the cases are going to be related to either accidents or DUI. Considering this we researched on different machine learning algorithms which don't focus on text similarity but text semantic similarity. Having semantic meaning of a description point towards the case we can know what case it is, what the accused did and under what punishment it categorizes. There are many text semantic similarity algorithms present out there but since our problem is critical we need most efficient and accurate result giving the algorithm to solve our cases. We narrowed down two algorithms which can give us results accurate to what expected, which is using biRNN MaLASTM Siamese networks and using cosine distance. Difference between the above two algorithms is only about how the distance between the two sentences is carried out, using cosine vector distance or Manhattan distance.

Extraction of semantic meaning from a simple or complex sentence is an important task when developing applications that deal with semantic parsing. A decent application should have the ability to gauge the meaning of a sentence with minimum errors. While selecting the algorithm we first looked into basic Recurrent Neural Network(RNN), however it is observed to have a vanishing gradient problem over large input. In the attempt to develop such an application, a model which includes Siamese Neural Networks where Long Short Term Memory cell replaces neurons, is used. Though the addition of LSTM cells increasing the complexity of the model by introducing various new weights, it overcomes the problem of vanishing gradient, this is achieved by including an additional element of memory output for self-looping connections. LSTM cells are individual computing units with a key feature called cell state which is important in the modification of the input vector to produce the desired output. This vector is computed using the structures called gates, the gates that are involved are, namely, input gate, forget gate and output gate. These gates usually use a sigmoid function (F) to optionally allow the information to flow through to the cell, the values ranging from 0 to 1, where 0 means completely block the information coming in and 1 means to let it flow through completely.

The system accepts two sentences as an input, which are the FIR description, an explicit account of the incident and the Section Statement(already available with the system). These statements are pre-processed to generate a word vector of 300d and is embedded and then passed to the LSTM RNN network. The model developed has two similar LSTM RNN networks which process a given sentence in pairs to get the semantic numeric value of both the sentences in a pair. Our focus is to get the closest or equal semantic value of a pair (LSTMa = LSTMb). This LSTM model is able to learn the mapping of variable length sentences of dimension vectors. Each LSTM network takes a pre-processed sentence 'x' as an input. The pre-processed sentence is in its sequential word

vectors form, i.e. x1, x2,..., xt. This sentence representation in the form of sequential word vectors is then given to the LSTM network which then adjusts the hidden cell weights according to the sentence semantics using the gates. The final semantic value of the sentence will be the value of last hidden cell weight (ht) of the LSTM network. Then to calculate the similarity between the two semantic values of the input sentences we use the necessary function. Here we use Manhattan distance formula for function

$$g(hta, htb) = \exp(- |hta - htb|) \qquad (1)$$

falling in the range [0,1], hta, htb - semantic value of sentence 1 and 2 respectively. Lower the value more similar is the sentences. To enable our model to generalize beyond the limited vocabulary in training sets, we provide our LSTM model with pre-trained word-vectors which are trained on the external corpus. We use 300-dimensional word2vec embeddings which represent as word vector for each word in our input sentences. To demonstrate the use of it in a simple example, it can be shown as vec(queen) - vec(woman) + vec(man) = vec(king). Having a vast number of words representing word vectors we cover more than required vocabulary to achieve a semantic value of any sentence according to our goal concerned with the proposed application.

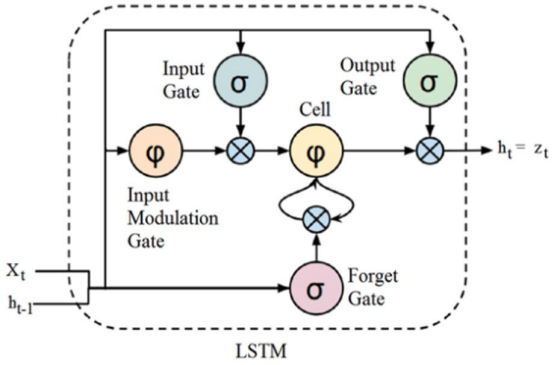

Figure 1. LSTM Cell

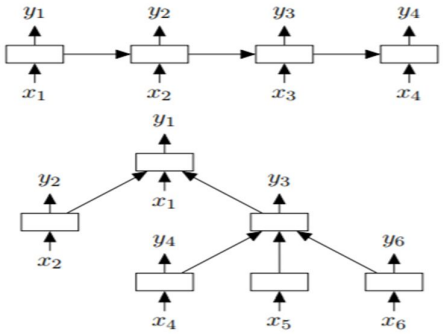

Figure 2. Semantic Tree

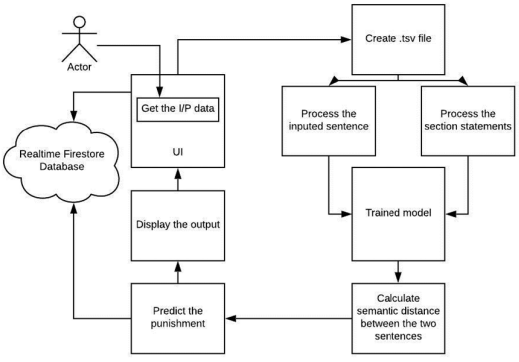

Figure 3. Block Diagram of the System

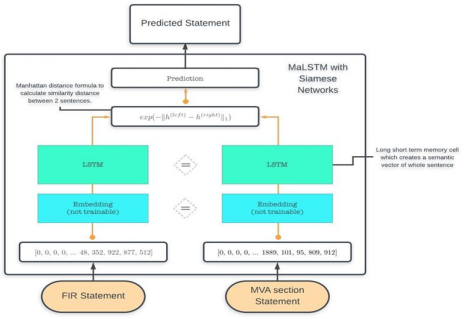

Figure 4. Architecture of the System

In the system developed, for the research purposes we have selected our domain to accommodate just two sections of MVA, namely, DUI and Accident related cases. Hence the outcome of the system is a verdict(punishment based on the Section being mapped).

IV. EXPERIMENTATION

The FIR being an official inquiry document mandatory for a legal procedure of criminal nature has an exhaustive collection of fields containing information regarding the pertinent offence/incident. However, resolving a case requires a concise set of parameters. This refining of data is done in the pre- processing stage of the application where relevant fields of the FIR are saved in our database which is accessed by our model to make the

calculations to determine the verdict. The fields that are crucial for giving a verdict in our application are FIR description, age, offenders history (sufficient for a simple case as driving under the influence of alcohol/narcotics, compensation in accidents). After extracting the pertinent fields from the FIRs, the application stored them in Google's cloud-hosted platform of Firebase. The details are processed by the model are explained in the sections to follow.

A. Creation of Database

For the training of the model the important fields of the FIRs are:

i. FIR description
ii. The statement of the mapped MVA Section (we have implemented it for 185 and 140)

The model creates a semantic vector for each FIR description and Section statement, then processes it to finally obtain the Manhattan distance between the vectors to determine whether the statement of the section matches the description of the FIR or not thereby making a decision so as to print the punishment statement of the section or not. Now for this to be successfully implemented the model requires a dataset with three fields, namely,

1) FIR description
2) The statement of the MVA Section
3) 0 or 1 (1 indicating the section statement and FIR description being a match and 0 indicating no direct relationship) As we obtained a hundred FIR documents which were either a match or not to the section (two in our case), the number of unique entries into our dataset is two hundred.

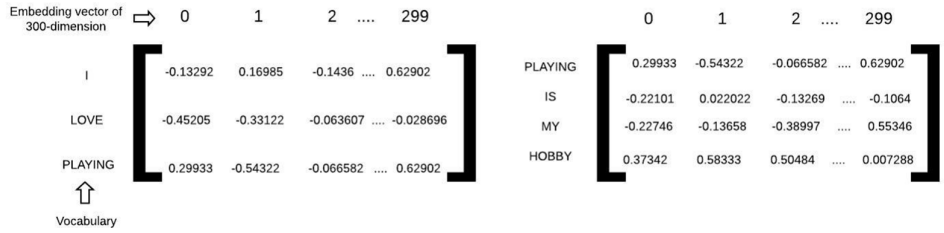

Figure 5. Embedding vectors of "I LOVE PLAYING" and "PLAYING IS MY HOBBY"

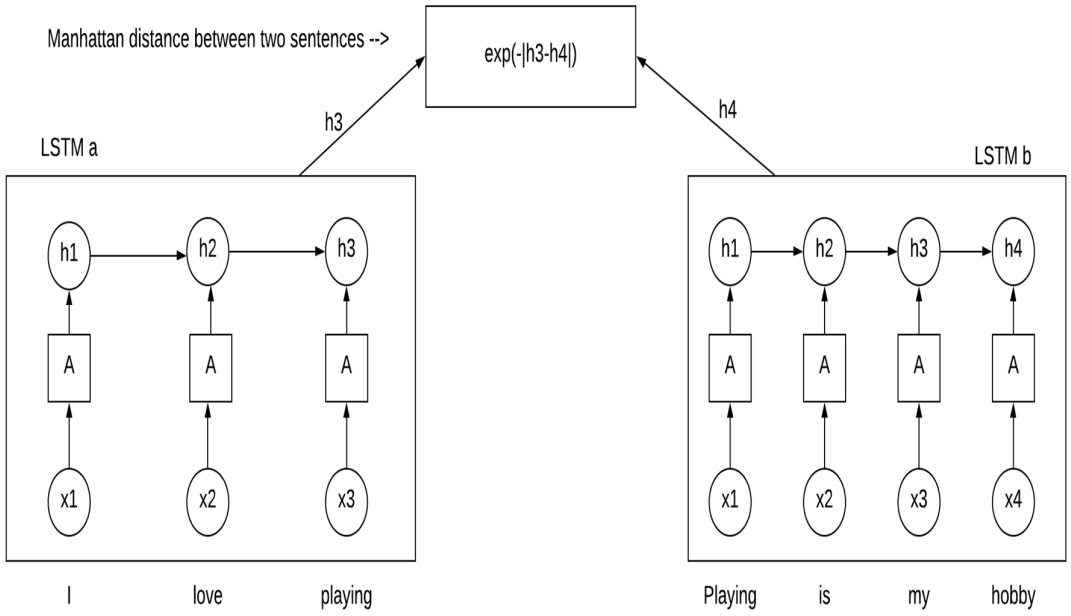

Figure 6. Simplified network diagram of the System with same example

B. Working of the System

As discussed in the methodology, the model needs two input sentences to check the similarity between them. The proposed application takes FIR description as an input and the accused details and also victim details, if any. The application stores it in our cloud based datastore for further use. The application then creates pairs of sentences such that the FIR description is mapped with each MVA act we are considering in a tsv file. To create diversity we have considered more MVA sections to pair it with but our end goal is to map the FIR description to either of two MVA sections i.e. Accidents or DUI. By doing this we can check how much semantically similar the given input FIR description is to Accident or DUI cases. For each pair we input it to our LSTM Siamese model to get the similarity between two sentences in a pair. For a similar pair final output is 1 else 0. Thus, for all pairs we get to know to which MVA section the current case belongs to. The application also goes one step further to check under which punishment the accused falls under by checking its correlation for subsections of the MVA section to which the input description is matched. The system follows the same method as above, to check the similarity between the input FIR description and all the considered MVA sections, to check under which subsection the current case falls into. After getting the final similarity result, the application checks for the punishment related to which section and respective subsection the FIR description is matched to, it displays the final verdict with the required fields such as FIR description, accused persons name, offense count, punishment.

C. Training of the Model

Before training the model we split the whole training dataset into a large part of train sets and a small part of development set to make sure the classifier is developed with some variations. The pre-trained 300- dimensional word2vector is loaded first as our dictionary over which the model is trained. The model is then initialized with 3 hidden layers of LSTM and 50 hidden units. The word embedding unit converts the given sentences into its word embeddings. These word embeddings of two sentences now are used as input to the two RNN network layers respectively. With the outputs of the last hidden state of the two network layers as semantic value of the two sentences the Manhattan distance is calculated between them using the exponential function, Manhattan distance given by:

$$d = \exp(-\|hta - htb\|) \qquad (2)$$

The model is trained using supervised learning, and so loss function takes input of the difference d of the two semantic values calculated and what actual output y should be i.e. 1(similar) or 0(not). Loss is calculated using the reduced sum of y*square(d). 0.5 is the threshold value considered, accordingly for distance less then 0.5 is considered as similar to each other whereas they are not similar otherwise. The value for similar sentence pair is 1 and for dissimilar it is assigned as 0. The accuracy is based on the mean of correct predictions evaluated.

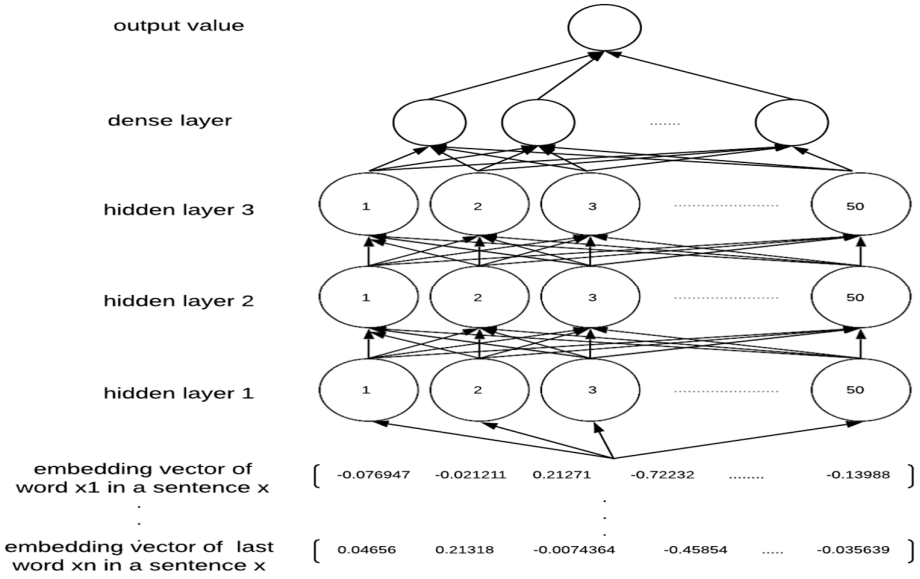

Figure 7. Three Hidden layers and the Dense layer

V. RESULTS

The System is implemented using Python PyQT for user interface and using MaLSTM Siamese Network for training and testing of the FIRs. It is observed to give an accuracy of about 93% while mapping the sections with the case statements. Along with providing the punishment, the system will also use a record of the history of the offender. The collated output will be displayed in a relevant file format. Confusion matrix is displayed to show our output mapping accuracy. The true values are plotted against the predicted values. According to the confusion matrix in Fig. 9, 97% of the cases that were not supposed to be mapped to a particular section were correctly not mapped to that section. 3% of the test cases were incorrectly mapped to a particular section. Likewise, 89% of the cases that were supposed to be mapped to a particular section, were mapped to that section, and 11% of the cases were incorrectly not mapped to that section.

The model was tested on 40 FIR sentences. We have used confusion matrix as a metric to measure the error in the prediction made by the proposed model for these sentences. As seen in the Fig. 9, a very low false positive and false negative predictions were made. According to Fig. 8, 18 out of 20 accident related cases were correctly mapped and 19 out of 20 DUI cases were correctly mapped.

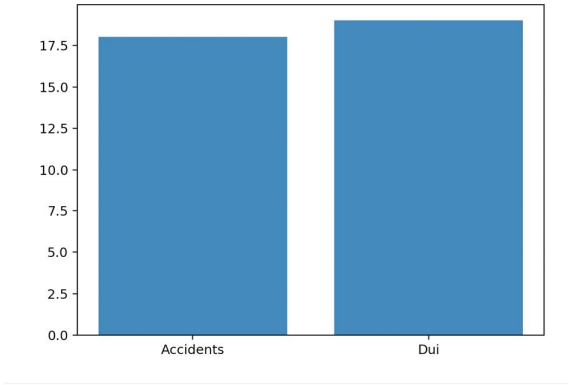

Figure 8. Results of testing performed on 40 FIRs

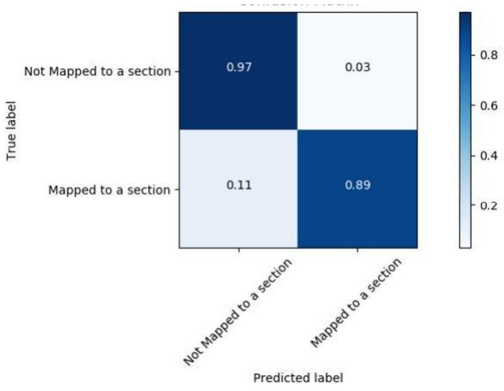

Figure 9. Confusion Matrix as per the testing results

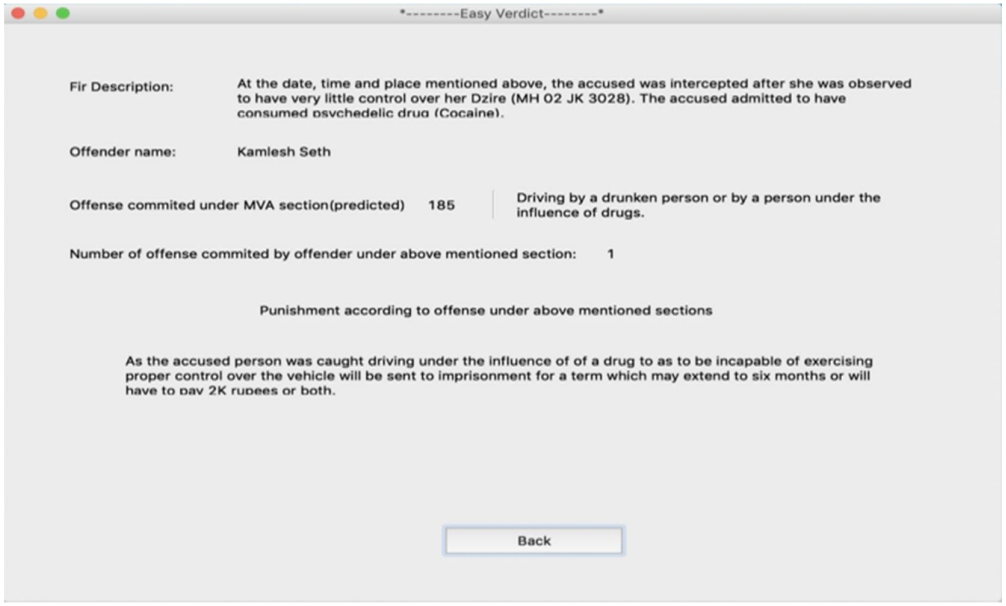

Figure 10. Verdict as per the offense

III. Conclusions

We have implemented the program with an accuracy of 93% of predicting the correct punishment for a case statement. The scope of this project can be extended to include more types of cases in the future, like thefts and violence related crimes. This concept could be used to give predictions on real time data of other cases as well. The scope of the project can also be extended to include cases with multiple interrelated crimes that happened in one scenario, or in a sequence of events caused by a common phenomenon.
There needs to be an involvement of a human, any person involved in judicial system, to check if the punishment given is correct, because even when the application gives the correct results the judiciary cannot afford to give a wrong decision even to a small number of cases.

References

[1] Thyagarajan Aditya, Jonas Mueller (2015). Siamese Recurrent Architectures for Learning Sentence Similarity published in www.mit.edu Journal, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

[2] Paul Neculoiu, Maarten Versteegh and Mihai Rotaru. 2016. Learning Text Similarity with Siamese Recurrent Networks in Proceedings of the 1st Workshop on Representation Learning for NLP, pages 148157.

[3] Henry Nassif, Mitra Mohtarami, James Glass. 2016. Learning Semantic Relatedness in Community Question Answering Using Neural Models in Proceedings of the 1st Workshop on Representation Learning for NLP, pages 137147

[4] Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 17351780. doi:10.1162/neco.1997.9.8.1735

[5] Marelli, M.; Bentivogli, L.; Baroni, M.; Bernardi, R.; Menini, S.; and Zamparelli, et al., R. 2014. SemEval-2014 Task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. SemEval 2014.

[6] Zhao, J.; Zhu, T. T.; and Lan, M. 2014. Ecnu: One stone two birds: Ensemble of heterogeneous measures for semantic relatedness and textual entailment. SemEval 2014.

[7] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed Representations of Words and Phrases and their Compositionality. NIPS 31113119.

[8] Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. ACL 15561566.

[9] He, H.; Gimpel, K.; and Lin, J. 2015. Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks. EMNLP 15761586.

[10] Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. Computer Vision and Pattern Recognition 1:539546.

[11] F. Javed, Q. Luo, M. McNair, F. Jacob, M. Zhao and T. S. Kang, "Carotene: A Job Title Classification System for the Online Recruitment Domain," *2015 IEEE First International Conference on Big Data Computing Service and Applications*, Redwood City, CA, 2015, pp. 286-293. doi: 10.1109/BigDataService.2015.61

[12] R. Pascanu, T. Mikolov, and Y. Bengio. 2013. On the difficulty of training recurrent neural networks. Journal of Machine Learning Research

[13] Neil Zeghidour, Gabriel Synnaeve, Maarten Versteegh, and Emmanuel Dupoux. 2015. A deep scattering spectrum - deep siamese network pipeline for unsupervised acoustic modeling. In IEEE International Conference on Acoustics, Speech and Signal Processing.

[14] Will Y Zou, Richard Socher, Daniel M Cer, and Christopher D Manning. 2013. Bilingual word embeddings for phrase-based machine translation. In EMNLP, pages 1393–1398.

[15] Sepp Hochreiter and Jurgen Schmidhuber. 1997. ¨ Long short-term memory. Neural computation, 9(8):1735–1780.

[16] Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In Similarity-Based Pattern Recognition, pages 84–92. Springer.