Deep Learning Algorithms Based Text Classifier

Arthi Venkataraman
BOS-Botworks
Wipro Technologies
Bangalore, India
arthi.venkat@wipro.com

***Abstract*: There exists a base classification system for classification of problem tickets in the Enterprise domain. Different deep learning algorithms (Gated Recursive Unit and Long Short Term Memory) were investigated for solving the classification problem. Experiments were conducted for different parameters and layers for these algorithms. Paper brings out the architectures tried, results obtained, our conclusions and way forward.**

Keywords— ***Natural Language Processing; Machine Learning; Classification; Accuracy; Deep Learning; Gated Recursive Units; Long Short Term Memory; Embedding Layer; Optimizer; Adagrad; Adam; AdaDelta; Ridge classifier;***

I. INTRODUCTION

Employees face a lot of issues across different functions in the organization like Payroll, Infrastructure, Facilities, Applications, etc. When employees raise a ticket they are traditionally asked to manually select a category for their ticket from a hierarchical menu driven interface. This leads to a lot of wrong selection of choice by end user. This in turn causes tickets to be raised in wrong bucket and delays the resolutions due to re-assignments.

We have built a system which accepts a Natural language text input from end user and automatically classifies the ticket in the right category. The challenges faced and technical steps needed to build such a system are described in detail in Arthi1 [2015] by same author. In Arthi2 [2015] techniques are discussed to further improve the accuracy of the classification using automated class level patterns. To further improve accuracy we investigated if we can apply deep learning techniques to build these classifiers. We have tried different deep learning algorithms and architectures.

In this paper we give a brief background of the classifier system we have built. We then describe our experiments using deep learning models. It brings out our findings based on these experiments. This is an industry applied research paper. These experiments are a work in progress with additional architecture variants still being tried.

II. BACKGROUND

A classifier system was designed which understands the natural language description entered by end user to automatically log the incident in the correct category. There is a working system based on an ensemble of machine learning algorithms. We want to investigate if some of the deep learning algorithms can be applied to further improve the accuracy. In this paper we describe our experiments with different deep learning algorithms and the results we have obtained.

As a premise for this paper we assume the necessary data cleaning and other accuracy improvement techniques as described in Arthi 1 [2015] are already applied. We focus on the different deep learning classification architectures we have tried and the conclusions we have arrived based on same.

III. RECENT WORKS

There is a lot of research addressing the techniques to build more accurate classification systems. Similarly in recent times there is a lot of research on deep learning specially applied to the textual domain. Few of the key ones are highlighted in this section.

Bengio [1994] discusses the difficulty of learning long term dependencies with gradient descent

In Hochreiter [1997] the LSTM (Long short term memory) network architecture was first discussed. It was shown how these networks could help overcome the vanishing and exploding gradients problem with normal recurrent networks.

In Evigeniy [2006] it has been discovered that use of a wider body of knowledge in the classification process enables a more intelligent and accurate classification.

In Pu Wang [2007] it has been found that the classification results with encyclopedia knowledge are much better than the baseline "bag of words" methods.

In Andreas [2008] the relationship between accuracy and number of features is evaluated.

In Feifan [2009] it is shown that simple unsupervised TFIDF approach performs reasonably well and the additional information from POS and sentence score helps keyword extraction

In Ariel [2009] it is shown that we can have large accuracy gains using automatically extracted training data at much lower cost.

In Mikolav [2010] it is proved that connectionist language models are superior to standard n-gram techniques.

IEEE Part Number: CFP16D66-USB IEEE ISBN: 978-1-5090-2398-1
Presented at International Conference on Applied and Theoretical Computing and Communication Technology, Bangalore, 2016

In Chuang [2010] novel techniques like shuffling datasets to create different decision boundaries in several ways, random subspace, and bootstrapping has been attempted. The smoothing-average method is found to result in higher classification accuracy.

In Smith [2011] it is shown that by identifying and removing instances which are misclassified the classifier accuracy increases.

Pascanu [2013] has shown that using deep networks outperform shallow networks in the language modelling tasks. In Graves [2013] a detailed study has been done for generating sequences by predicting one data point at a time using Long Short-term Memory recurrent neural net-works.

In Peng [2014] an improved feature weighting approach, is discussed which reflects the importance of the term in the positive and the negative training examples,

In Yanhong [2014] used word2vec for describing a document along. The resulting word2vec vector was clustered and a new label assigned to each cluster. Use of SVM based classification for these new labels gave good accuracy.

In Vishwanath [2014] a KNN based machine learning technique is applied to categorize documents.

In Shenkai [2014] a method for generating ensembles by explicitly maximizing classification accuracy and diversity of the ensemble together using a multi-objective evolutionary algorithm is discussed.

In Andrew [2015] a partitioned embedding neural network was proposed which improves upon word2vec by modelling orderings

In Arthi 1 [2015] techniques and solution for building a large scale accurate classifier is discussed.

In Arthi 2 [2015] techniques and solutions are discussed to augment a classifier ensemble using automatically generated class level patterns which work more accurately specially for class which have less data.

IV. PROPOSED SOLUTION APPROACH

This section will describe the key blocks of the classifier system in a brief manner. This system is elaborated in Arthi 1 [2015].
Fig. 1 represents the main high level modules of the overall solution:

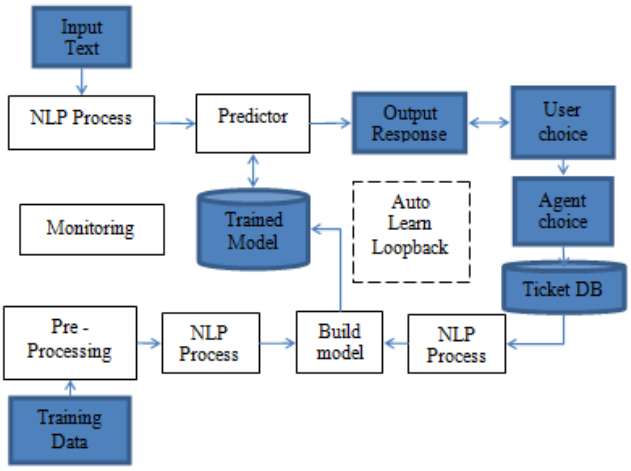


Fig. 1. Overall Solution Diagram

A. *Pre-Processing*

Here data is made suitable for building accurate models by techniques such as active learning, similar class mergers, stop word removal and random over-sampling.

B. *NLP Processing*

In this step the unstructured text in tokenized. The words are then converted to word2vec vectors. Alternately a One-hot encoding can be used or Glove vectors can be used.

C. *Classifier*

The actual deep learning model architecture is chosen and parameter for same is fine-tuned as part of this step. An iterative process of selecting an algorithm, choosing the architecture, fine tuning the parameters, building the model and evaluating same is done till the required accuracy is achieved. Once the required accuracy is reached the algorithm is selected, the parameters baselined and the model is built for the training data.

D. *Predictor*

Prediction module predicts the correct classes for the given input using the models stored in the trained model store. The prediction model is responsible for running the different ensemble of models, combining results from different models and scoring the same.

E. *Automated Learning Loopback*

This module is responsible for ensuring the accuracy of the system increases with use. Whenever a ticket is closed in a class which is different from the originating class it is fed back as training data for the closure class. The ticket database is

IEEE Part Number: CFP16D66-USB IEEE ISBN: 978-1-5090-2398-1
Presented at International Conference on Applied and Theoretical Computing and Communication Technology, Bangalore, 2016

monitored regularly for such cases. These tickets are then fed through the NLP process block and then to the Build model module to re-build the models.

V. Experimental Setup

A. Data set

For testing our algorithms we used tickets falling under different functional categories. The source of these tickets is the historical tickets which have been logged by the users of the ticketing system. These tickets are stored in the MySQL database and are retrieved using suitable queries. This includes leave, applications, payroll, , etc.

Below are the details of the data corpus we have used for our experiments:

Total classes – 66

Data size (Number of rows) - 217,000

TABLE 1: Data composition of Training Set

Number of Tickets in Bucket	***Number of Classes***
< 200	9.00
200 - 500	13.00
500 - 1000	8.00
1000 - 10000	30.00
> 10000	6.00
Total Classes	66.00

B. Deep Learning Algorithms Tried

The section gives a background of the deep learning algorithms we have experimented on.

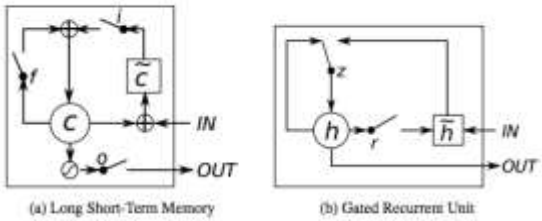
1) Long Short Term Memory Network (LSTM)

The Long Short Term Memory network architecture is a neural network where the regular network units are replaced with LSTM blocks. LSTM blocks contain additional gates in the form of input gates, forget gates and output gates. They also have a memory unit to store states. LSTM architectures help in learning long term dependencies and help overcome the vanishing and exploding gradients problem. This is achieved by the learning capability provided by the memory units as well as the presence of the different gates. A gate will trigger only if the input crosses a certain threshold allowing the unit to selectively receive inputs, selectively forget inputs as well as selectively pass inputs to the output units.

2) Gated Reurrent Units (GRU)

This is a variant of the LSTM. Here the input and forget gates are combined into a single gate. There is no separate memory unit. The hidden unit also doubles into the memory unit. The resulting architecture is more efficient.

Fig. 2 represents an image of the LSTM and GRU as published by http://deeplearning4j.org/.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 1: Illustration of (a) LSTM and (b) gated recurrent units. (a) i, f and o are the input, forget and output gates, respectively. c and $\tilde{c}$ denote the memory cell and the new memory cell content. (b) r and z are the reset and update gates, and h and $\tilde{h}$ are the activation and the candidate activation.

Fig. 2: Diagramatic Representation of LSTM and GRU (from DeepLearning4j)

C. Data Pre-processing

The data set is split into training and testing sets. The data set is split using the 70:30 percentage split ratio with 70% of tickets used of training. All data is passed through an Embedding layer where the sentence inputs are converted into a sequence of shape [batch size, sequence length, embedding size]. The embedding size chosen was 50.

D. Experimental Configurations

Our approach was to try different deep learning architectures for building the classifier model. Models were trained for the different architectural configurations using the pre-processed data. The test data was used for prediction using the different identified architectures. The output will be measured for each of the configurations using the identified metrics. A comparison will then be arrived for the performance across different configurations.

Based on experimentation with different learning rates. A low learning rate made the convergence very slow. After testing the learning rate of .01 was chosen.

All the experiments below were tried with the learning rate of .01.

Fig. 3 brings out a high level view of the generic architecture pipeline.

Fig. 3: High level view of the architecture pipeline

Different architectures with their variations tried are described below.

IEEE Part Number: CFP16D66-USB IEEE ISBN: 978-1-5090-2398-1
Presented at International Conference on Applied and Theoretical Computing and Communication Technology, Bangalore, 2016

Configuration 1 – Single LSTM Layers

A single LSTM layer was used. A learning rate of .01 was used. The Adagrad optimizer was used. The number of steps to run over the data was set to 100.

Configuration 2 – 4 LSTM layers

Four LSTM layers are sequenced one after the other in this architecture. A learning rate of .01 was used. The Adagrad optimizer was used. The number of steps to run over the data was set to 900000

Configuration 3 – Single GRU Layers

A single GRU layer was used. A learning rate of .01 was used. The Adagrad optimizer was used. The number of steps to run over the data was set to 100.

Configuration 4 – 4 GRU layers

Four GRU layers are sequenced one after the other in this architecture. A learning rate of .01 was used. The SGD optimizer and the Adagrad optimizer was used. The number of steps to run over the data was set to 800000 for SGD and Adagrad was set to 500000.

Baseline Configuration – Ridge Classifier

This configuration is used to have a base comparison model against a standard non deep learning based technique. We have used the Ridge classifier as the baseline model. This is one of the models as tried in Arthi 1 [2015].

TABLE 2 ACCURACY ACROSS CONFIGURATIONS

Single LSTM	***Single GRU***	***4 Layer LSTM***	***4 Layer GRU Adagrad***	***4 Layer GRU SGD***	***Baseline Ridge***
52%	55%	69.4%	65%	72.3%	75%

VI. EVALUATION METRICS

We selected the accuracy as the overall evaluation metrics. This is calculated by the number of correct prediction Vs the total number of predictions made.

VII. RESULTS

Each of the data sets was sent through the different configurations.

Figure 4 shows the result of predicting using the different configurations for the application group.

From fig. 1 we can see following:

- Using the baseline Ridge Classifier gives better results than any of the deep learning techniques
- The GRU performed slightly better than the LSTM for this task
- Using multiple layers compared to a single layer gave better result
- Setting learning rate to any value less than .01 led to very slow convergence
- The loss was continuously reducing with the training cycles. Hence if training was continued for more cycles further improvement in accuracy can be expected
- From run time perspective the time taken to train the deep learning models was much more than the time taken to train the baseline ridge classifier.
- Training was done on non GPU configuration. Hence training with a GPU configuration should speed up the training time
- While the accuracy of predictions of the deep learning models increased with the number of training steps. However increasing the number of steps also further increased the training time.

Fig. 4: Accuracy score across configurations

VIII. CONCLUSION

It is a daunting task to create a working prediction system with good level of accuracy and performance for a large scale live system while supporting diverse types of classes. Text classification in real life is a challenging issue. In this paper we have applied the deep learning algorithms LSTM and GRU to the text classification task. For the data set we have and the configurations we have tried the baseline Ridge classifier is better suited for live use due to its much shorter training time and higher accuracy. However we see good potential in the deep learning models for larger data sets. We would also need to further test the results with variations in the different parameters. We will also need to test for more number of test cycles till the loss converges. Further testing should be done on a GPU system. The GRU seems to perform a bit better than the

IEEE Part Number: CFP16D66-USB IEEE ISBN: 978-1-5090-2398-1
Presented at International Conference on Applied and Theoretical Computing and Communication Technology, Bangalore, 2016

LSTM. Also multi-layered models perform better than corresponding single layer models. Increasing the number of training cycles significantly improved the accuracy of results.

IX. FUTURE WORK

This experiment is work in progress and the results are our initial results. There is lot of work still left to do on same.

We need to investigate more accurate classification using solutions like word2vec. Use of word2vec to create the embedding representation can bring out the semantic dependencies in the sentence and should lead to more accurate classification.

We plan to vary the parameters for each of the different models. We need to try with different learning rates. We will experiment with different optimization techniques like Rmsprop, Sgd, Adamdelta and Adamax. We will also try pre-processing technique like Batch normalization. We will have to investigate the role of Drop out layer and how it will impact the accuracy. There is also a plan to investigate the use convolutional neural networks for text classification. Experiment need to be conducted with different number of layers.

The data itself we have tried experiments on in a small subset of the complete data. We need to try the experiments on larger and different data sets. The aim would be to arrive at the correct parameter which suit each of the different types of data.

Experiments will also be conducted for longer training cycles as well as on dedicated GPU machine.

ACKNOWLEDGMENT

I give my heartfelt thanks to my organization Wipro who has enabled me to build this system. They have also provided the needed support to make this presentation. I would like to thank my family and specially my spouse and parents for all their support. I would like to thank Google and additional contributors for the Tensorflow library as well as Francois Chollet and additional contributors for the keras library using which these experiments were run.

REFERENCES

[1] Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." Neural Networks, IEEE Transactions on 5.2 (1994): 157-166

[2] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[3] Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge by Evgeniy Gabrilovich and Shaul Markovitch Department of Computer Science Technion—Israel Institute of Technology, 32000 Haifa, Israel {gabr, shaulm}@cs.technion.ac.il (Unpublished)

[4] "Improving Text Classification by Using Encyclopedia Knowledge by Pu Wang ; Peking Univ., Beijing ; Jian Hu ; Hua-Jun Zeng ; Lijun Chen. Data Mining, 2007 ICDM 2007. Seventh IEEE International Conference on Data Mining, Pages 332 -341

[5] On the Relationship between Feature Selection and Classification Accuracy Andreas G. K. Janecek andreas.janecek@univie.ac.at Wilfried N. Gansterer wilfried.gansterer@univie.ac.at, FSDM, 2008

[6] Unsupervised approaches for automatic keyword extraction using meeting transcripts. By Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL '09, pages 620–628, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[7] Improving Classification Accuracy Using Automatically Extracted Training Data by Ariel Fuxman, Anitha Kannan and others. Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining Pages 1145-1154

[8] Mikolov, Tomas et al. "Recurrent neural network based language model." INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010 1 Jan. 2010: 1045-104

[9] Novel techniques to increase classification accuracy in machine learning by Tzu-Cheng Chuang, Purdue University (Unpublished)

[10] Improving Classification Accuracy by Identifying and Removing Instances that Should Be Misclassified Michael R. Smith and Tony Martinez. IJCNN, the 2011 International Joint Conference on Neural Networks, Pages 2690 – 2697

[11] Pascanu, Razvan et al. "How to construct deep recurrent neural networks." arXiv preprint arXiv: 1312.6026 (2013).

[12] Graves, Alex. "Generating sequences with recurrent neural networks." arXiv preprint arXiv: 1308.0850 (2013)

[13] A New Study Based on Word2vec and Cluster for Document Categorization by Yanhong YUAN, Liming HE, Li PENG, Zhixing HUANG. Journal of Computational Information Systems 10: 21 (2014) 9301–9308

[14] PU text classification enhanced by term frequency–inverse document frequency-improved weighting by Tao Peng, Lu Liu, Wanli Zuo. Concurrency and Computation: Practice and Experience. Volume 26, Issue 3 10 March 2014 Pages 728–741

[15] KNN based machine learning approach for text and document mining. International Journal of Database Theory and Application by Vishwanath Bijalwan1, Vinay Kumar2, Pinki Kumari3 and Jordan Pascual. International Journal of Database Theory and Application Vol.7, No.1 (2014), pp.61-70

[16] Generating diverse and accurate classifier ensembles using multi-objective optimization by Shenkai Gu, Yaochu Jin. Computational Intelligence in Multi-Criteria Decision-Making (MCDM), 2014 IEEE Symposium.. Pages 9 – 15

[17] Cho, Kyunghyun et al. "Learning phrase representations using rnn encoder-decoder for statistical machine translation." arXiv preprint arXiv: 1406.1078 (2014)

[18] Modeling Order in Neural Word Embedding's at Scale, Andrew Trask, David Gilmore, Matthew Russell, Proceedings of the 32'nd International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37

[19] Machine learning techniques for building a large scale production ready classifier by Arthi Venkataraman. Proceedings of the Second International Conference on Data Mining, Internet Computing, and Big Data, Reduit, Mauritius 2015

[20] Augmenting a Classifier Ensemble with Automatically generated class level patterns for Higher Accuracy, Arthi Venkataraman, 2015 IEEE Conference on Technologies and Application of Artificial Intelligence held in Taiwan

IEEE Part Number: CFP16D66-USB IEEE ISBN: 978-1-5090-2398-1
Presented at International Conference on Applied and Theoretical Computing and Communication Technology, Bangalore, 2016