ON-LSTM : An Empirical Analysis

Aditya Aggarwal(aaggarwal301), Anirudh Choudhary(achoudhary46), Mihir Mavalankar(mmavalankar3)

Spring 2019 CS 7643 Deep Learning: Final Report

Georgia Tech

Abstract

In this project we attempt to apply a novel approach to a well known problems in natural language processing, namely langua modelling and text classification. The paper on Ordered Neurons takes a different approach in this domain with the underly hypothesis being that natural language is hierarchically structured: smaller units (e.g. noun phrases) are nested within larger ur (e.g. clauses). While the different units of an LSTM can learn to track information at different time scales, the standard architect does not impose this sort of strict hierarchy. In particular, we aim to evaluate a novel RNN framework proposed recently in IC 2019, 'ON-LSTM (https://arxiv.org/pdf/1810.09536.pdf)' which explicitly integrates the latent tree structure into RNN and focuses modeling the hierarchy of constituents. The paper claims that it outperforms state-of-the-art LSTM based language model like AW LSTM (https://arxiv.org/pdf/1708.02182.pdf) which we intend to verify by evaluating on Toxic Comment Classificat (https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data) dataset hosted on Kaggle.

Background and Related Work

Natural language has a sequential overt form as spoken and written, but the underlying structure of language is not stri sequential. Sentences have additional structure beyond word ordering which chain-structured LSTM is unable to capture. T structure is usually tree-like: smaller units (e.g.,noun phrases) are nested within larger units (e.g., clauses). While Recurrent neu networks (RNNs) have proven highly effective at the task of language modeling, RNNs explicitly impose a chain structure on t data which seem at odds with the latent non-sequential structure of language and poses several difficulties for the processing natural language data with deep learning methods. This leads to their limited ability in capturing long-term dependencies a achieveing good generalization on conversational tasks. Ordered Neuron based LSTM presents a novel method for introducin syntax-oriented inductive bias without explicitly injecting linguistic knowledge. It builds upon the existing sequential LSTM framew without explicitly modifying the network structure. This is a major area of research in representation learning for NLP.

Developing deep neural network techniques that can leverage latent tree structure to form better representations of natural langua sentences have received a great deal of attention in recent years. Many attempts at language modelling suffer from inducing triv structure (e.g., a left-branching or right-branching tree (Williams et al., 2018)), or encounter difficulties in training caused by learn branching policies with Reinforcement Learning (RL) (Yogatama et al., 2016). Furthermore, some methods are relatively complex implement and train, like the PRPN model proposed in Shen et al. (2017).

ON-LSTM brings recurrent neural networks closer to performing tree-like composition operations by including novel gating mechanism and activation function. In previous studies, following are 3 key networks which have made an attempt at incorporating tree structue into LSTMs:

- **Tree-LSTMs** : TreeLSTMs oncorporate a tree network topology into LSTMs and have been shown to outperform conventional LSTM on sentiment analysis tasks[7]. However, they have access to a pre-defined true structure and model the tree-structure using parsing information from an external parser.
- **Clockwork RNN**: It segments the hidden state of a RNN by updating at different time-scales. However, it makes a strong assumption about the regularity of the hierarchy involved in modelling the data.
- **Parsing-Reading-Predict Networks (PRPN)**: It attempts to perform parsing by using self-attention to compose previous state and the range of attention is controlled by a learnt “syntactic distance”. This distance corresponds to the depth of the parse tre albeit at an added complexity.

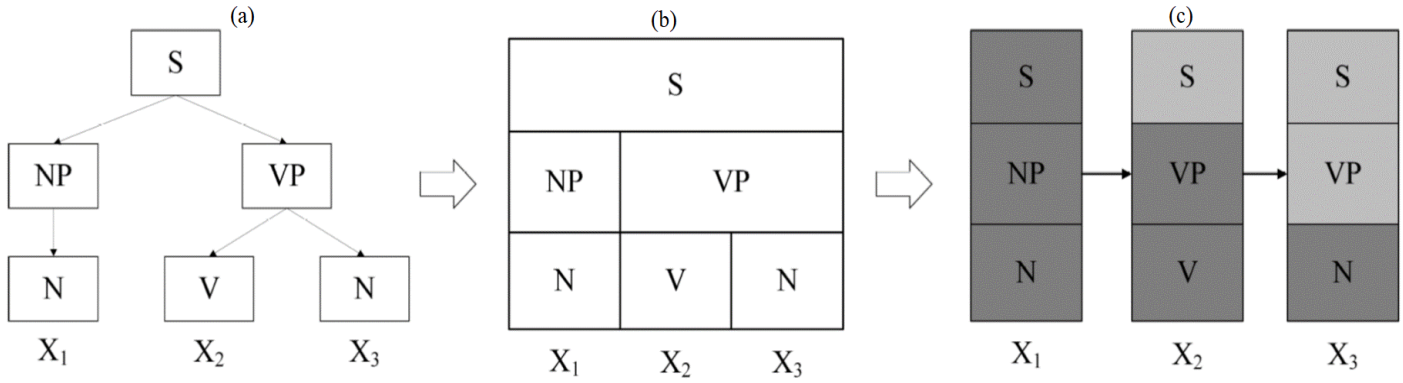


Figure 1: The relationship between a constituency parse tree and an ON-LSTM. Given a sequence of tokens (x1; x2; x3), their constituency-based parse tree is illustrated in (a). (b) provides a block view of the tree structure, where S and VP node strides across more than one time step. The representation for high-ranking nodes should be relatively consistent across multiple time steps. (c) visualization of the ratio of updated neurons for each group of neurons at each time step. At each time step, given the input word, darker grey blocks are completely updated, lighter grey blocks are partially updated. The three groups of neurons have different update frequencies. Higher groups update less frequently and lower groups update more frequently.

In this project, we aim to evaluate a novel RNN framework proposed recently in ICLR 2019, 'ON-LSTM' which presents substantially new way of introducing a syntax-oriented latent structure into sentence-level models without explicitly injecting linguis knowledge. The model proposes the concept of 'Ordered Neurons' that learns the differentiation of the life cycle of information sto inside each neuron: high-ranking neurons will store long-term information which is kept for a large number of steps, while lo ranking neurons will store short-term information that can be rapidly forgotten. The paper claims that it outperforms current state- the-art language modeling model like AWD-LSTM which we intend to verify by evaluating on Toxic Comment Classificat challenge hosted on Kaggle. This is a major topic of research in representation learning for NLP, and opens new possibilities research opportunities. Moreover using the Kaggle Toxic Commnet Classification Challenge will give us a good idea on how t new approach works on interesting real world problems.

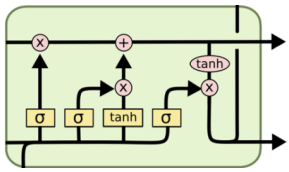
Approach

The approach given in the original paper is that given a sequence of tokens x1; : : : ; xT governed by a latent tree structure as sho in Figure 1(a), the motive is to infer the latent structure from the observed tokens while computing the hidden state h(t) for each ti step t. For each time step, h(t) should ideally contain information about all the nodes on the path between the current leaf node x and the root S. In Figure 1(c), we illustrate how h(t) would contain information about all the constituents that include the curr token xt even if those are only partially observed.

To achieve this, the paper presents a new RNN unit, ON-LSTM (“Ordered Neurons LSTM”). Ordered Neuron incorporates t master gates corresponding to hidden and input gates. The task of master gates is to control information flow in the original hidd and input gates. These gates are binary and the starting point of 1s is decided using the cumulative softmax function (cumax()) Master Forget Gate, the presence of smaller number of zeros indicate that information at lower word level are discarded while hig number of zeros indicate that complete information at all levels of constituency parse tree are discarded. In Master Input Gate, lov number of 1s indicate that current input information is a low level information and can be updated frequently while higher numbe 1s indicate that current information corresponds to a higher level node and needs to be retained over long term.

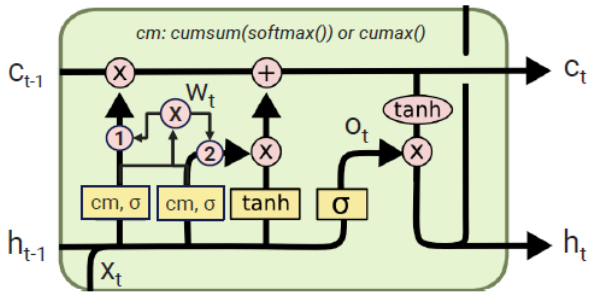
Figure 2: Functioning of Master Gates in Ordered Neuron

The new model shares a similar architecture with the conventional LSTM model:



$$
\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\
\hat{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \tanh(c_t)
\end{aligned}
$$

Figure 3: Internal diagram of LSTM cell along with the corresponding equations for a forward pass



$$
\begin{aligned}
\tilde{f}_t &= cumax(W_{\tilde{f}} x_t + U_{\tilde{f}} h_{t-1} + b_{\tilde{f}}) \\
\tilde{i}_t &= 1 - cumax(W_{\tilde{i}} x_t + U_{\tilde{i}} h_{t-1} + b_{\tilde{i}}) \\
w_t &= \tilde{f}_t \circ \tilde{i}_t \\
\hat{f}_t &= f_t \circ w_t + (\tilde{f}_t - w_t) \quad (1) \\
\hat{i}_t &= i_t \circ w_t + (\tilde{i}_t - w_t) \quad (2) \\
c_t &= \hat{f}_t \circ c_{t-1} + \hat{i}_t \circ \hat{c}_t
\end{aligned}
$$

Figure 4: Internal diagram of ON-LSTM cell along with the master gate functions for a forward pass. Both cumax() and sigmoid() functions are applied to create f(t); f'(t) & i(t);i'(t)

The only difference with the standard LSTM is that it excludes the update function for cell state c(t) and replaces it with a new upd rule. Since the gates in the standard LSTM act independently on each neuron, this paper proposes to make a gate for each neu dependent on others by enforcing an order of neuron update.

Our contributions to the existing methodology are the following:

1. ON-LSTM claims to achieve better performance on language modeling tasks by introducing inductive bias and separately allocating hidden state neurons with long and short term information. Their model achieves slightly better performance than baseline AWD-LSTM which has been the state of the art for language classification tasks. We verified this claim by replicating their results on Penn TreeBank dataset. While ON-LSTM doesn't significantly improve in terms of accuracy over AWD-LSTM, ON-LSTM performs better in sentence parsing with the middle layer incorporating longer term information. Logically, its more sound than AWD-LSTM which is an efficient engineered solution.
2. Recently, ULMFiT (https://arxiv.org/abs/1801.06146), a transfer learning framework which achieves high accuracy on text classification tasks has been proposed. Their framework is based on AWD-LSTM and utilizes pretrained language model from wiki103 corpus. ULMFiT involves layer-wise training for fine-tuning a pretrained language model on new classification task. Ol LSTM has been shown to be more generalizable over tasks with varying text sequence lengths and involving long term dependencies. We integrated ON-LSTM model into ULMFiT's existing framework in FastAI library to perform language modeli and classification. We also evaluate ULMFiT (https://arxiv.org/abs/1801.06146) using its pretrained AWD-LSTM model agains our classification models trained from scratch on toxic comment classification task.

Experiments and Results

Based on the way we have approached this project project we have two main tasks. First was to recreate the results of the origi ON LSTM paper on a language modeling task on the Penn Tree Bank Dataset and the second one is to classify the toxic comme in the kaggle jigsaw toxic comment classification challenge. In the first part, we measure the perplexity of the language model, wh is in essence a probability distribution over sentence, phrases, sequence of words, etc. In the second part, for classification task, success metric is the accuracy, precision, recall and F1 score of our 4 networks built and trained from scratch. We also evalu them against a classification model based on ULMFiT by comparing performance on the test dataset from Kaggle challenge.

The Penn Tree Bank Dataset has One million words of 1989 Wall Street Journal material annotated in Treebank II style, a sn sample of ATIS-3 material annotated in Treebank II style and a fully tagged version of the Brown Corpus. It also has switchbo tagged text and Brown Corpus parsed text.

The dataset in the Kaggle Toxic comment classification task at about 2 million comments from the the Civil Comments platfo which shutdown in 2017. Jigsaw sponsored this effort and extended annotation of this data by human raters for various to conversational attributes. In the data supplied for this competition, the text of the individual comment is found in the comment_t column. Each comment in Training dataset has a toxicity label (target), and models should predict the target toxicity for the T data. This attribute (and all others) are fractional values which represent the fraction of human raters who believed the attrib applied to the given comment. For evaluation, test set examples with target >= 0.5 will be considered to be in the positive cla (toxic). The dataset is highly imbalanced with toxic comments contributing only 10% of overall comments. To balance the datas we select equal number of positive and negative comments (100k each) for training the models.

With this project we want to analyse the performance of the ON-LSTM model and compare it to different other language mode The experimental plan was divided into the following parts: -

1. Firstly we reproduced part of the experiment conducted in the original paper for ON-LSTM, in particular the experiment where measures its performance against AWD-LSTM as that is widely used for state-of-the-art language modeling. We ran this experiment on the Penn TreeBank dataset mentioned in the paper and check to what extent our results match the empirical results highlighted in the paper.
2. Next, we developed 4 LSTM models based using AWD-LSTM and ON-LSTM architectures but with sight modifications. ON-LSTM follows the archiecture of AWD-LSTM (3 layers with 400,1150,400 activations) with almost equivalent parameters. We used a 2 layer simplified architecture (400 and 256 activations) to be consistent across LSTM and AWD/ONLSTM networks. Also, the last 400 dimension layer in AWD-LSTM and ON-LSTM are primarily targeted towards language modeling tasks with word embedding or size 400. We compared the performance of ON-LSTM and AWD-LSTM with conventional LSTM on Toxic Comment Classification task
3. Finally, we trained a language model using ULMFiT's transfer learning framework using AWD-LSTM with pretrained wiki103 based word embeddings and language model. This was to verify whether transfer learning could be a viable alternative in toxi comment classification given the nature of conversation dataset and the dataset of which the pre-trained model was built(wikipedia) is very different. Although ULMFiT based model had lower accuracy on train dataset, it generalized better on t unseen test dataset from Kaggle and generated a higher AUC Score. Motivated bu this finding, we integrated ON-LSTM mode in fastai's ULMFIT codebase for language modeling and classification tasks in addition to other language modeling framework already present - AWD-LSTM and Transformer XL.

Hyperparameter tuning

- Classification models trained from scratch (Table 1) : For LSTM and bidirectional LSTM, we used a learning rate of 1e-3 for training. For ON-LSTM and AWD-LSTM, we experimented with 6 learning rates ranging from 1e-4 to 10 and found learning ra of 0.5 to be the best. Their original implementations also utilize a high learning rate of 30 for language modeling.
- For ULMFiT based model, we evaluated various learning rates and used a learning rate of 2e-2(last layer training) and 1e-3(complete language model training). For classification task, we used learnign rate of 3e-4.

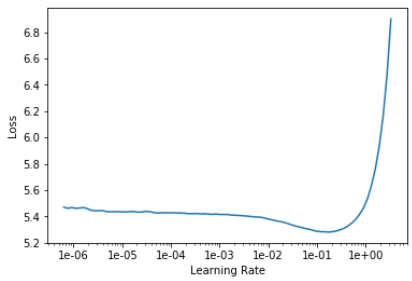


Figure 5: ULMFiT Language Model - Loss vs Learning Rate

The success of this project can be evaluated by comparing the reproduced results with the one claimed in the paper. Also, the kaggle competition rank can also measure the efficacy of the model. The results, both quantitative and qualitative for this task are follows: -

Quantitative results

Results for all the models on the toxic comment classification task

Models	Accuracy	Precision	Recall	F1 Score
Vanilla LSTM	87.96	86.93	89.26	88.08
Bi-Directional LSTM	87.20	85.02	90.21	87.54
AWD LSTM	89.05	88.53	89.65	89.09
ON-LSTM	89.30	88.32	90.50	89.39

Results for all the models for language modelling task on the Penn TreeBank Dataset

Models	Claimed Perplexity		Reproduced Perplexity	
	Validation	Test	Validation	Test
AWD LSTM: 3 layer (tied)	60.00	57.30	61.41	58.89
ON-LSTM: 3 layer (tied)	58.29	56.17	57.98	56.12

Results for classification using ULMFiT's transfer learning on pre-trained language model (AWD-LSTM with wiki103 based embeddings)

Task	Accuracy	AUC Score
ULMFiT Language Modeling	30.32	-
ULMFiT Classification Model(Train)	85.29	-
ULMFiT Classification Model(Test)	85.20	0.916
ON-LSTM Classification Model developed from scratch (Test)	89.30	0.909

Qualitative results

For the qualitative analysis we could observe from the Figure 5 that toxic comments which were correctly tagged by ON-LSTM missed by AWD-LSTM had equitable distribution across sequence length but comments correctly tagged by AWD-LSTM but miss by ON-LSTM are concentrated in smaller sequence length. This is in line with the behaviour reported in ON-LSTM's publication t it starts performing better than LSTM on sequences with length >=8 and thus should be better at capturing longer te dependencies in language. (Figure 6)

Analysis

The results that we have achieved show that that the original hypothesis ON-LSTM's are indeed better at such as capturing lo term dependencies and achieving good generalization. The model's results on unsupervised constituency parsing result shows t the ON-LSTM induces the latent structure of natural language in a way that is coherent with human expert annotation. The induct bias also enables ON-LSTM to achieve good performance on language modeling, long-term dependency, and logical inferer tasks. We can see that the results achieved by us are very similar to the ones reported in the original paper without any fine tur hyper-parameter tuning.

To get a better understanding of the qualitative results discussed in the previous section, we have made a density plot visualizat below. On the x-axis are the number of words in the comments and the y-axis is the score for them. These are all instances comments that were toxic from the toxic comments dataset. The orange area shows the instances that the AWD LSTM got right were misclassified by the ON-LSTM and the blue region shows the instance that the ON-LSTM got right but were misclassified the AWD-LSTM.

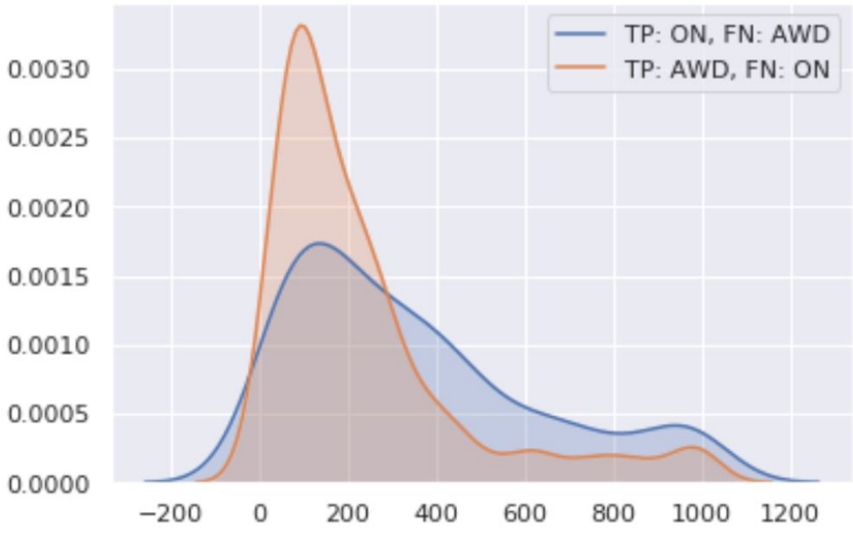


Figure 6: True Positive Rate vs Sequence Length

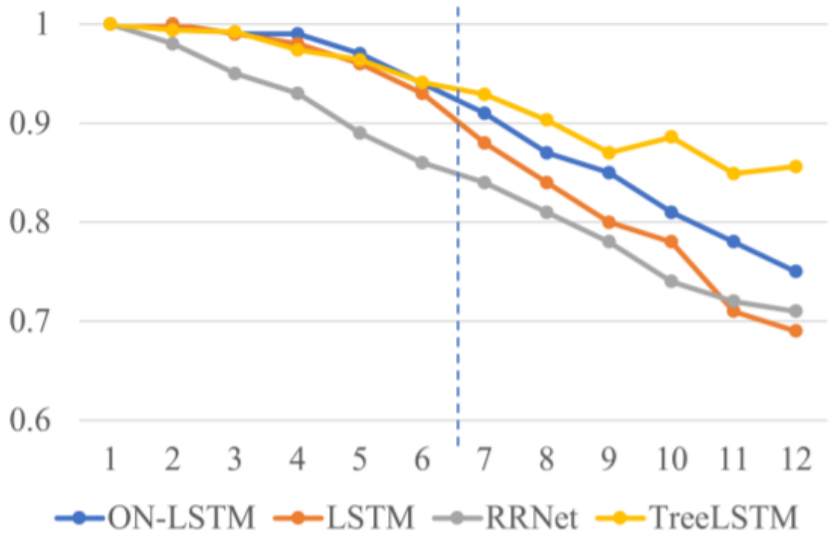


Figure 7: Test accuracy of ON-LSTM vs other networks on logical inference tasks with varying sentence length(x-axis)

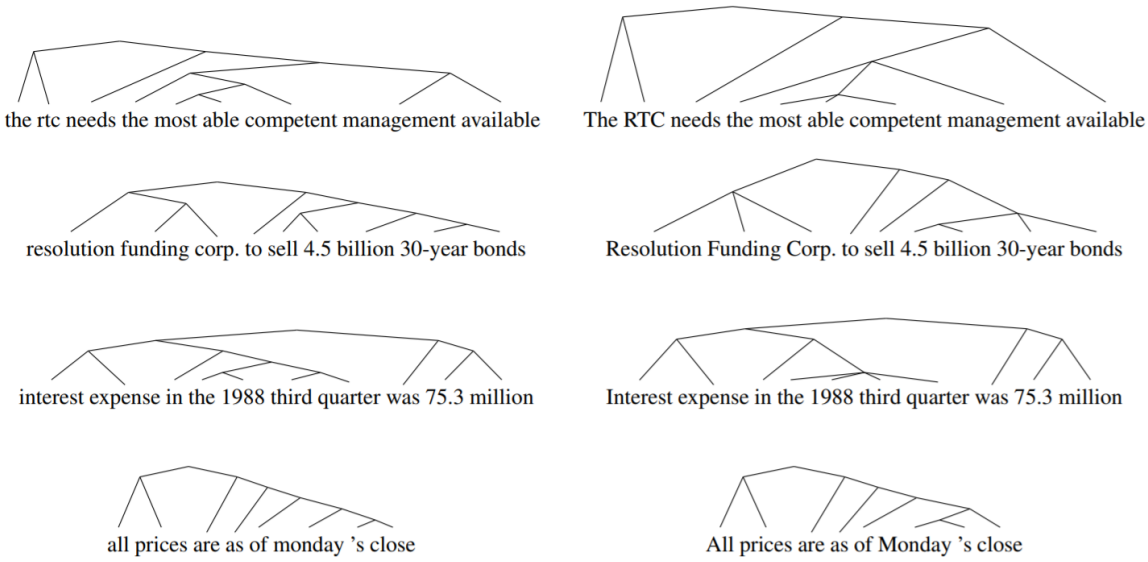


Figure 8: Sample parses from ON-LSTM (right: ground truth, left:ON-LSTM's 2nd layer)

For the plot we can see that for longer sentences ON-LSTM is clearly better at capturing meaning and classifying them corre while for shorter ones the AWD method is better. Since the test set contains sentences of various lengths which are unobserv during training, we find that ON-LSTM provides better generalization and robustness toward longer sentences than previous mode This is in line with ON-LSTM paper's claim that ON-LSTM can provide strong results for phrase detection, including ADJP (adject phrases), PP (prepositional phrases), and NP (noun phrases) and generates very accurate parses (Figure 8). This feature co benefit many downstream tasks, like question-answering, named entity recognition, co-reference resolution, etc.

Our experiment highlights that ON-LSTM is a promising language modeling framework which presents a novel way of introducin syntax-oriented latent structure into sentence-level models. Empirical results show that ON-LSTM outperforms AWD-LSTM sligh albeit with a higher execution time. In the future, we plan to thoroughly test the ON-LSTM implementation in ULMFiT's framew and leverage it for faster learning and better generalization.

Challenges

Some of the challenges that we encountered while working on this project were: -
Problems anticipated

1. Fine-tuning required to replicate empirical results shown in ON-LSTM/AWD-LSTM papers
2. Ease of modification of ULMFiT framework to incorporate ON-LSTM model
3. Data pre-processing challenges in toxicity challenge's conversations
4. Resource constraints for training ULMFiT framework
5. Fine-tuning ULMFiT framework

Problems encountered

1. We are getting results which are close to the empirical results in AWD-LSTM and ON-LSTM paper. However, the training of O LSTM is quite slow and hyperparameter tuning for classification task took significant time.
2. Modifications which were required in FastAI's ULMFiT framework to incorporate ON-LSTM which proved to be more time consuming than anticipated, particularly the weight loading and parameter specification.
3. ULMFiT fine-tuning and pre-processing is memory intensive and hence was problematic to get up and running for ON-LSTM which is quite slow than AWD-LSTM even when we were training it from scratch with a 2 layer network.

Distribution of work among team members

Name	Description of Work
Aditya Aggarwal	Replicate AWD-LSTM model as mentioned in the paper and incorporate it to the existing language modeling architecture for text classification task and compare the results of ON-LSTM and AWD-LSTM based models
Anirudh Choudhary	Explore existing language modeling frameworks, implement ULMFiT based language and classification models and incorporate ON-LSTM into ULMFiT framework. Also contributed to poster design, framework creation and model diagrams.
Mihir Mavalankar	Replicate ON-LSTM model as mentioned in the paper, validate the results of the paper and incorporate it to the existi language modelling architecture for text classification task.

References

1. AdinaWilliams, Andrew Drozdov*, and Samuel R Bowman. Do latent tree learning models identify meaningful structure in sentences? Transactions of the Association of Computational Linguistics, 6:253–267, 2018.
2. Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. Learning to compose words into sentences with reinforcement learning. arXiv preprint arXiv:1611.09100,2016.
3. Yikang Shen, Zhouhan Lin, Chin-Wei Huang, and Aaron Courville. Neural language modeling by jointly learning syntax and lexicon. arXiv preprint arXiv:1711.02013, 2017.
4. Yikang Shen, Shawn Tan, Alessandro Sordoni and Aaron Courville. Ordered Neurons: Integrating Tree Structures Into Recurrent Neural Networks. arXiv preprint arXiv:1810.09536, 2018.
5. Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and Optimizing LSTM Language Models. arXiv prepr arXiv:1708.02182, 2017.
6. Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In Proceedings of ACL 2018
7. Kai Sheng Tai, Richard Socher, Christopher D Manning. Improved Semantic Representations From Tree-Structured Long Sho Term Memory Networks. In Proceedings of ACL 2015