# Temporal Convolution Network Approach to Insincere Question Classification

Andre Douglas[1]

***Abstract*— Determining the sincerity of a question or sentence in general is an active area of research in the field of Sentiment Analysis. This is particularly difficult as unlike conventional sentiment analysis which seeks to determine if a string's emotional content is positive, neutral or negative; insincerity requires parallel analysis of tone-neutrality, disparaging or inflammatory content, baseless claims, and sexual content [1]. This nuanced, simultaneous survey of multiple characteristics increases the likelihood of misclassification. We seek to develop a reliable and scaleable Artificial Neural Network based on a Temporal Convolution Network Architecture to improved the accuracy of classification of question sincerity from a Quora dataset. Temporal Convolution Networks have a number of strong characteristics which combine simplicity, autoregressive prediction and very long memory [2]. These characteristics are are attractive in this application as we require no information leakage and preservation of sequence length. Our source code is publicly available at `https://github.com/jamieandre/w266-final-project`.**

## I. INTRODUCTION

An existential problem for any major website today is how to handle toxic and divisive content. Quora wants to tackle this problem head-on to keep their platform a place where users can feel safe sharing their knowledge with the world.An insincere question is defined as a question intended to make a statement rather than look for helpful answers. Some characteristics that can signify that a question is insincere: has an exaggerated tone to underscore a point about a group of people, is rhetorical and meant to imply a statement about a group of people; suggests a discriminatory idea against a protected class of people, or seeks confirmation of a stereotype, makes disparaging attacks/insults against a specific person or group of people, based on an outlandish premise about a group of people, Disparages against a characteristic that is not fixable and not measurable, Based on false information, or contains absurd assumptions, uses sexual content (incest, bestiality, pedophilia) for shock value, and not to seek genuine answers.

Temporal Convolution networks are attractive for this problem as their architectures are suited to detect time invariant nuances in the sentences. The TCN is designed from two basic principles: the convolutions are causal, meaning that there is no information leakage from future to past. The architecture can take a sequence of any length and map it to an output sequence of the same length just as with an RNN – another strong candidate architecture for this problem [2]. To achieve the first point, the TCN uses causal convolutions, i.e., convolutions where an output at time t is convolved only with elements from time t and earlier in the previous layer. To accomplish the second point, the TCN uses a 1D fully-convolutional network architecture, where each hidden layer is the same length as the input layer [3].

The dataset provided by Qoura is of size 1306122 x 3 for training and 56370 x 2. The data fields on the dataset include in this order: the unique identifier for each question, an integer, question text, a unscrubed string and a target integer: 1 for insincere question and 0 for a sincere question [4].

A series of embeddings are provided for use: GoogleNews-vectors-negative300, glove.840B.300d, paragram_300_sl999 and wiki-news-300d-1M. In the final model only glove and wiki-news word embeddings were utilized. Both embeddings were blended into an embedding matrix for use in the TCN [4].

TCNs can be build to have very long effective history sizes, which means they have the ability to look very far into the past to make a prediction. To this end, a combination of very deep networks augmented with residual layers and dilated convolutions are deployed.

## II. ALGORITHM

Simple causal convolutions have the disadvantage to only look back at history with size linear in the depth of the network, i.e. the receptive field grows linearly with every additional layer. To circumvent this fact, this architecture employs dilated convolutions that enable an exponentially large receptive field. More formally, for an input sequence $\boldsymbol{x} \in \boldsymbol{R^T}$ and a filter $h : \{0, ..., k-1\} \rightarrow R$ he dilated convolution operation $H$ on element $x$ of the sequence is defined as

$$H(x) = (\boldsymbol{x} \cdot_d h)(x) = \Sigma_{i=0}^{k-1} f(i) \boldsymbol{x}_{s-d \cdot i}$$

Where $d = 2^v$ is the dilation factor, with $v$ the level of the network and $k$ is the filter size. The term $s - d \cdot i$ accounts for the direction of the past. Dilation is equivalent to introducing a fixed step between every two adjacent filter taps.

Using larger dilation enables an output at the top level to represent a wider range of inputs, thus effectively expanding the receptive field of a CNN. There are two ways to increase the receptive field of a TCN: choosing lager filter sizes $k$ and increasing the dilation factor $d$ since the effective history of one layer is $(k-1)d$.

[1]Douglas is a MIDS Candidate at the University of California, Berkeley. Email: andred@berkeley.edu

[2]Scott Rushton is a Reader in the Department of Research and Technology, Stormloop Technologies, 75240. Email: scott@stormlooptech.com

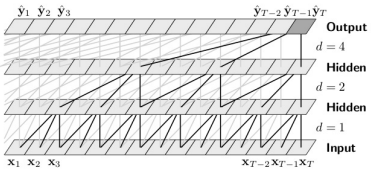


**Fig. 1:** Dilated Convolution Process Flow from time $T \rightarrow T_N$ for 3 layers

Another architectural element of a TCN are residual connections. In place of a convolutional layer, TCNs employ a generic residual module. Each residual block contains a branch leading out to a series of transformations $F$, whose outputs are added to the input $boldsymbolx$of the block.

$$\sigma = Activation(x + F(x))$$

This effectively allows layers to learn modifications to the identity mapping rather than the entire transformation, which has been shown to benefit deep neural networks. Especially for very deep networks stabilization becomes important, for example, in the case where the prediction depends on a large history size with a high-dimensional input sequence.

$$\hat{\boldsymbol{z}}^{(i)} = (\hat{z_1}^{(i)}, ..., \hat{z_T}^{(i)}$$

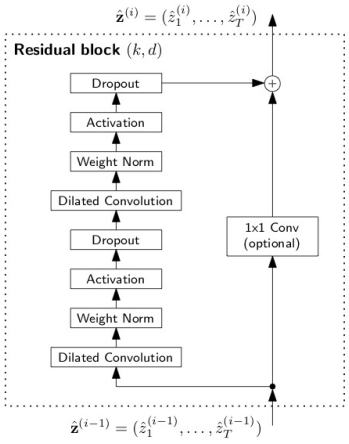


**Fig. 2:** Residual Block Dropout in R1

## III. Experiments

To invistagate the ability of TCNS to classify insincere questions and its robustness to hyperperameter choices, we performed a series of toy experiments with networks trained to provided Kaggle dataset.

**Data pipeline:** In order, the empty rows of the dataset were filled with filler test "_##_". A canonical tokenizer was used to convert the text to a sequencer of tokens, adhering to the training and testing dataset sizes. The sentences were padded to a predefined max length of 50 which was arbitrarily determined. The data was shuffled using a random seed and permuted along the length of the training set.

**Algorithm 1** Temporal Convolution Network. Sensible parameters are $d = [1, 2, 4]$ and $k = [64, 128]$.

```
Initialize running params, d and k
Convolve features in R1 for filter size k
while Training is not finished do
    Infer residual block, x.
    Activate convolved features by Wavenet
    Calculate 1D Spatial Dropout at dropout rate of 0.0
    if Spatial Dropout and Dialations are Present then
        Add layer for Spatial Drop Out
    else
        Activate residue using ReLu
    end if
end while
```

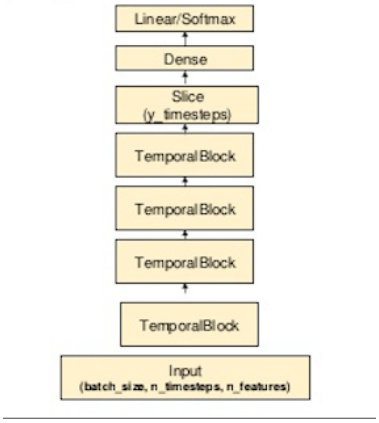


**Fig. 3:** Convolutional textsampling network with three skip-2 residual blocks.

**Architecture:** Text was passed through the convolutional network in fig. 3. Each convolutional layer is followed by Softmax[5] activation, except the last.

**Initialization:** All weights were Xavier[6] initialized. Biases were zero initialized.

**Learning policy:** ADAM optimization was used with the hyperparameters recommended in [7] and a base learning rate of 1/1280 for 100000 iterations. The learning rate was constant in Dialation 1, 4, 16 experiments and decreased to 1/12800 after 54687 iterations in Dialation 64 experiments. Networks were trained to minimize mean squared (loss) or quartic errors between test and training set.

## IV. Discussion

TCN model is deliberately kept simple, combining some of the best practices of modern convolutional architectures. Therefore, it can serve as a convenient but powerful starting point when dealing with sequential data.

TCNs can be build to have very long effective history sizes, which means they have the ability to look very far into the past to make a prediction. To this end, a combination of very deep networks augmented with residual layers and dilated convolutions are deployed [2].

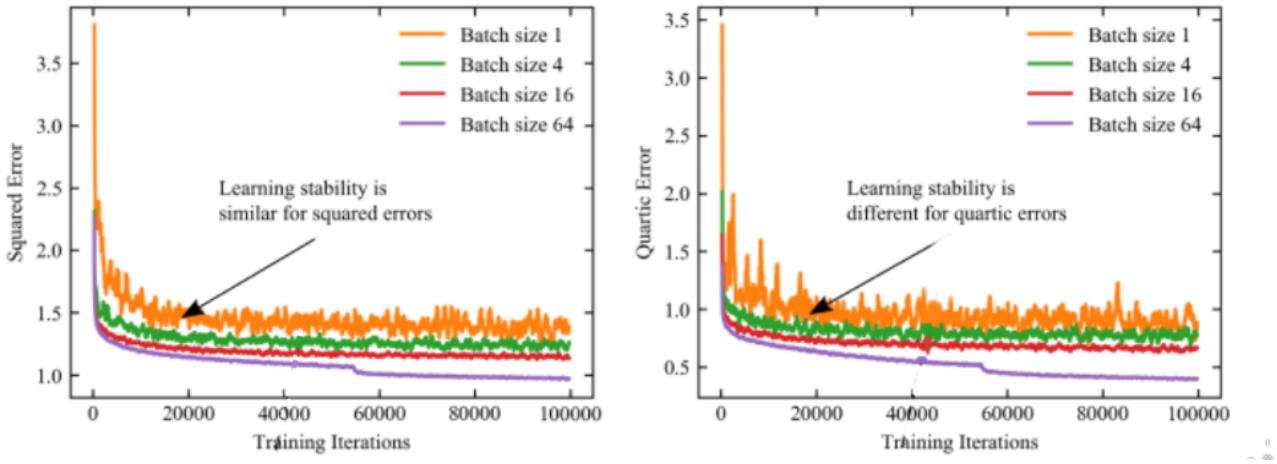


**Fig. 4:** Unclipped learning curves for text classification with Dialations 1, 4, 16 and 64 with and without adaptive learning rate clipping of losses to 3 standard deviations above their running means. Training is more stable for squared errors than quartic errors.

Squared Errors

| | Dialation 1 | | Dialation 4 | | Dialation 16 | | Dialation 64 | |
|---|---|---|---|---|---|---|---|---|
| Threshold | Mean | Std Dev | Mean | Std Dev | Mean | Std Dev | Mean | Std Dev |
| 2 | 5.35 | 0.048 | 3.56 | 0.016 | 4.58 | 0.010 | - | - |
| 3 | 5.32 | 0.054 | 3.46 | 0.029 | 4.58 | 0.004 | 3.90 | 0.013 |
| 4 | 5.36 | 0.048 | 3.17 | 0.017 | 4.58 | 0.007 | 3.89 | 0.016 |

Quartic Errors

| | Dialation 1 | | Dialation 4 | | Dialation 16 | | Dialation 64 | |
|---|---|---|---|---|---|---|---|---|
| Threshold | Mean | Std Dev | Mean | Std Dev | Mean | Std Dev | Mean | Std Dev |
| 2 | 8.54 | 0.084 | 3.02 | 0.023 | 2.60 | 0.012 | 1.65 | 0.011 |
| 3 | 8.59 | 0.055 | 3.08 | 0.024 | 2.61 | 0.014 | 1.58 | 0.016 |
| 4 | 8.61 | 0.054 | 3.13 | 0.023 | 2.64 | 0.016 | 1.57 | 0.016 |

**TABLE I:** Mean square losses 2, 3, 4 running standard deviations above their running means for Dialations 1, 4, 16 and 64. Means and standard deviations are multiplied by 100.

The TCN architecture appears not only to be more accurate than canonical recurrent networks such as LSTMs and GRUs, it also contains the following properties: Parallelism: Unlike in RNNs where the predictions for later time steps must wait for their predecessors to complete, convolutions can be calculated in parallel because the same filter is used in each layer. Therefore, in both training and evaluation, a long input sequence can be processed as a whole, instead of sequentially as in RNNs. Flexible receptive field size: The receptive field size can be changed in multiple ways. For instance, stacking more dilated convolutional layers, using larger dilation factors, or increasing the filter size are all viable options. Thus, TCNs afford better control of the model's memory size, and are easy to adapt to different domains [1].

Low memory requirement for training: Especially in the case of a long input sequence, LSTMs and GRUs can easily use up a lot of memory to store the partial results for their multiple cell gates. In TCNs, however, the filters are shared across a layer, with the back-propagation path depending only on the network depth. As dialations increase in magnitude, we see a decrease in mean squared error.

In this specific case, the model performed fairly with a strong f1 score max of 0.6517.

## V. Conclusions

We have developed TCN to classify quora questions as sincere or insincere. Our experiments show that TCN achieves convergence with low loses for this classification task.

## VI. Source Code

Source code for TCN experiments and a TensorFlow[3] implementation of ALRC is available at https://github.com/jamieandre/w266-final-project.

## References

[1] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," tech. rep., Citeseer, 2009.

[2] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems*, pp. 901–909, 2016.
[3] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning.," in *OSDI*, vol. 16, pp. 265–283, 2016.
[4] Quora, "Quora insincere questions classification," 2016.
[5] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.
[6] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.
[7] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

## VII. Acknowledgements

This research was funded by Stormloop Technologies grant EP/N035437/1.