CS 482/682 Final Project Report

**Text Categorization**

Cong Gao, Zhaohao Fu, Liujiang Yan

**Abstract:** Text categorization is a fundamental task in the area of natural language processing. In this report, we introduce several learning-based approaches, utilizing different representation for text, such as word-embedding based recurrent convolutional neural network and character level convolutional neural network. We compare the performance of different methods, adopting our own variants of small kernel size and skipping layer connection, on multiple public datasets as well as our own test dataset.

1. **Problem Statement**

Text categorization is an essential task in natural language processing with various applications such as information filtering, sentiment analysis, and text-based searching. The goal is to identify the subject from the original text input. Compared with computer vision problems, one key difference is that the representation of text is character based instead of numerical value based, which restricts the direct deployment from those well successful image classification methods. Previous works have been done to explore the representation for text that could be utilized for machine learning method. In this project, we mainly refer to two representations and corresponding approaches: word based representation and character based representation.

To demonstrate and compare the effectiveness, performance, and efficiency of the proposed methods, we perform the experiments with the following datasets: *Twitter Comments*, *UCI news*, and *AG news*. See the following description for detailed information about each of the datasets.

**Twitter Comments:** This original dataset contains twitter comments from Hillary Clinton and Donald Trump in 2016.01 to 2016.09. We also collect the data by mining their tweets up to date as our test set. We use this dataset as a baseline of performance.

**UCI News:** This dataset contains English news abstract of multiple major topics. We use this dataset for tuning the hyperparameters and architectures.

**AG News:** This dataset contains title and description of news articles from more than two thousand news sources. We will use it to evaluate the performance at final step.

1. **Methods**
2. **Word-level Recurrent Convolutional Neural Networks (Word-RCNN)**

**Word Embedding** This method utilizes word embedding for representation, which converts each word to a real-value vector and enables us to measure the semantics relevance from the distance between vectors. In our implementation, we use pre-trained model from Stanford NLP group [2].

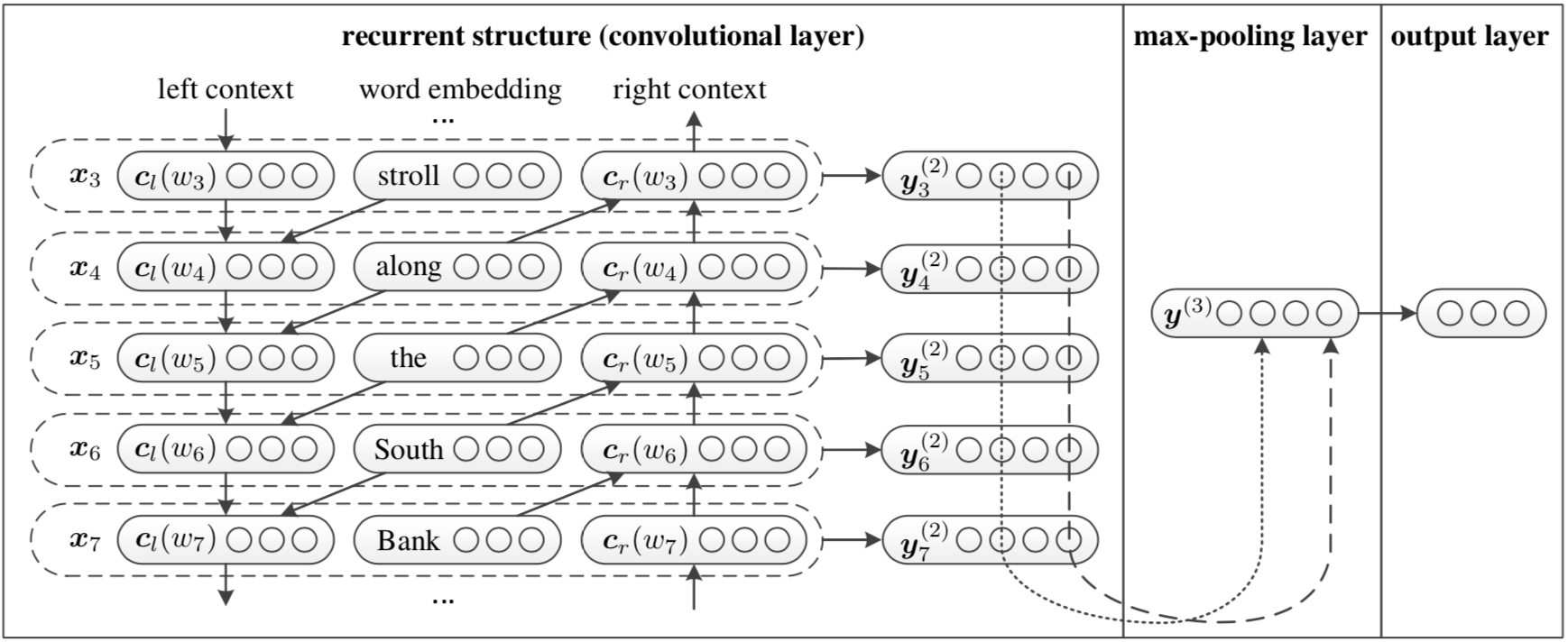


Fig 1. Architecture of the RCNN [1]

**Architecture** Figure 1. shows the architecture of the RCNN, which consists of several components. The first one is a bidirectional recurrent network, where the sequence input is the sequence of word embedding given by the text, and the hidden variables represent left and right contextual information correspondingly. Then, we stack the corresponding left and right contextual hidden variables with the input and form a feature vector. Then a fully connected layer connects the feature vector to a hidden layer and a max pooling layer is performed to capture the max responses at each position. The output is same size with the classes’ number and a log softmax activation function is performed to get the final output. Since it is a classification problem and the final activation is log softmax, we use negative log likelihood as our loss function.

**Variants**

1. **Zero-Padding**: The number of words per sentence is various therefore the length of representations are variant, leading to difficulty in using mini-batch training. We introduce a variant in representation by adding zero padding in the end to a fixed length. Note that we perform maxpooling at hidden layer, and the activation function from outputs of RNN to hidden layer is originally **tanh**, butzero padding might change the max response. Therefore, we correspondingly change the activation function to **ReLU**.
2. **Deep Text Learning**: The original network connects the outputs of RNN to final prediction by a shallow connection and it could be enhanced to a deep connection by adding 1d convolutional layers and max pooling layers.
3. **Character-level Convolutional Neural Networks (Character-CNN)**

**Character quantization** The representation here is character level. For each given text, it forms a fixed size binary matrix that each column is one hot vector size 1-of- with one for the character’s index in a given alphabet and zero otherwise; and the total row number is fixed to , any character exceeding length is ignored. Thus, the sequence of characters is transformed to a sequence of such sized vectors with fixed length .

**Architecture** Figure. 2 presents the architecture of Character-level CNN, consisting of convolutional layers, max pooling layers and fully connected layers. As discussed above, the final layer is a log softmax for multiple class classification task, and we use negative log likelihood as our loss function.

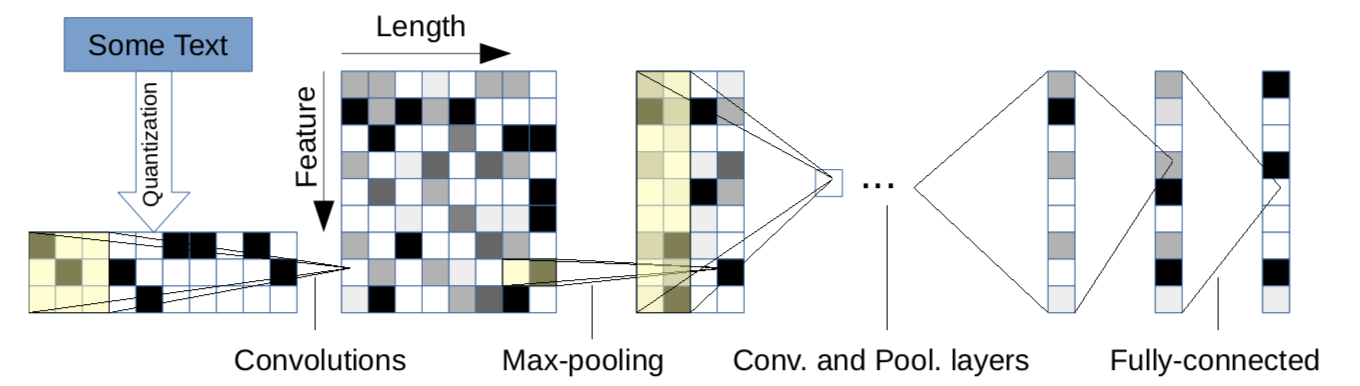


Fig 2. Architecture of the Character-level CNN [3]

**Variants**

1. **Batch Normalization:** Some of our dataset have short length in input, and might lead to gradient vanishing given this deep network. Therefore we add batch normalization in each conv layer.
2. **Kernel Size**: Inspired by VGG net, a composition of two conv layers of kernel size as 3 have same reception field of one conv layer of kernel size 7, but has more nonlinearity. Therefore, we replace the large kernel conv layer with composition of small kernel conv layers, but keep the pooling layers.
3. **Skip Layer Connection**: The other approach to apply deep net for simple / small data is to perform deep residual learning. We introduce the skip layer connection as basic block, and we also remove the fully connected layer to reduce the number of parameters.
4. **Results**

The experiments are performed on a local machine, MacBook Pro (15-inch, Early 2013, quad core 3th Intel Core CPU). The whole pipeline is implemented on PyTorch and we decouple the code into components: configuration, dataset, models, etc. To evaluate the proposed methods’ performance, we first shuffle the dataset and spilt each one to 80% as training and 20% as validation. We finally test the performance on AG news by splitting to 60% as training, 20% as validation and 20% as test. To evaluate the accuracy, we simply count the ratio of correctly classified samples.

**Tweets Comments**

As we introduce, this dataset is relatively small, and we use it to demonstrate whether the proposed architecture is sufficient for this task. For the following we show each method’s loss plot and accuracy plot. We also perform data mining to collect the up-to-date twitter comments as our test dataset.

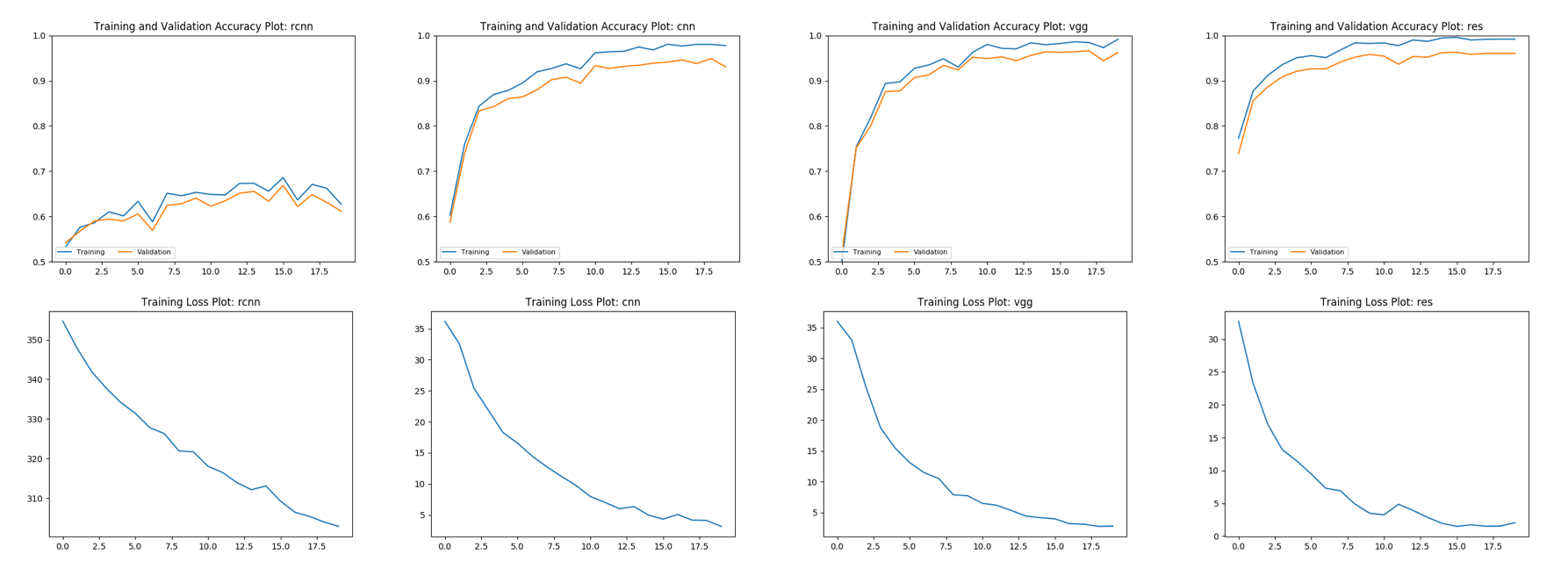


Fig 3. Accuracy and Loss Plot on Tweets Dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Word  RCNN | Character  CNN | VGG-Type  CNN | ResNet-Type  CNN |
| Validation Acc | 0.65 | 0.93 | 0.96 | 0.96 |
| Test Acc | 0.617 | 0.767 | 0.796 | 0.801 |

Table 1. Accuracy on Tweets Dataset

As the results show, character level CNN and its variants all achieve more than 90% classification accuracy in this dataset, and the variants of small kernel and skipping layer connection lead to about 4% accuracy improvement. The original RCNN method takes too much time to train due to single sample batch, therefore we only perform experiment by zero padding and use fixed length of 10 words and batch size of 10. Through the plot we see that, this approach does not achieve as good result as char CNN, which might result from the zero padding.

**UCI news**

The UCI news dataset is relatively large and has multiple classes, therefore we use it to demonstrate whether the proposed architecture can scale and achieve comparative accuracy. Through this scaled dataset, we shall see that, the introduction of small kernel and skip layer connection achieve significant performance improvement in aspect of classification accuracy.

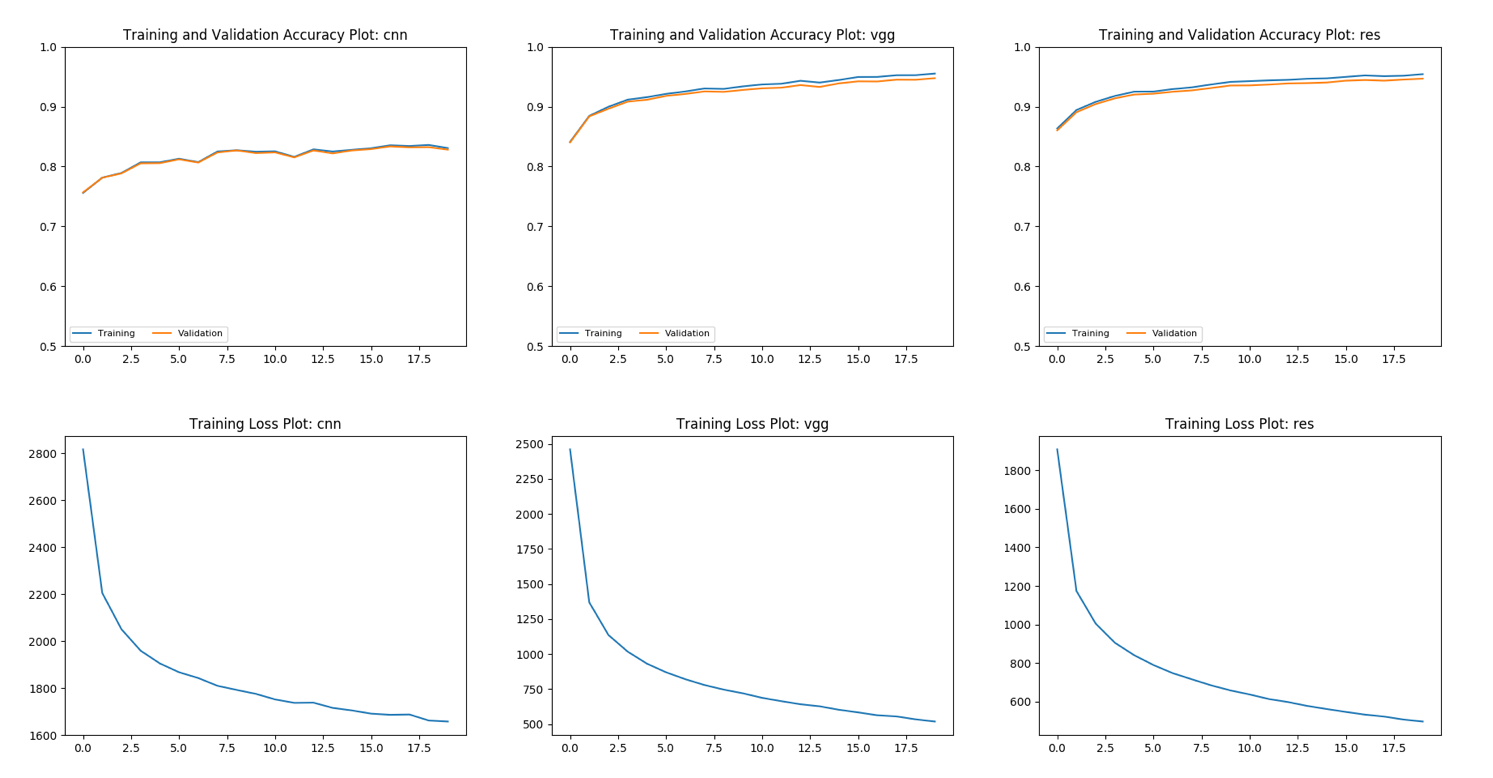


Fig 4. Accuracy on UCI news dataset

|  |  |  |  |
| --- | --- | --- | --- |
|  | Character CNN | VGG-Type CNN | ResNet-Type CNN |
| Accuracy | 0.828 | 0.947 | 0.946 |

Table 2. Accuracy on Tweets Dataset

**AG news**

We use AG news dataset to compare each method’s performance, by using same training data and validation data, and we test on a separated dataset. The results agree with the observations given above. We also compare to the result listed in the paper, noting that the epochs is 5,000 for their training while we only use 50, we still achieve comparatively similar accuracy.

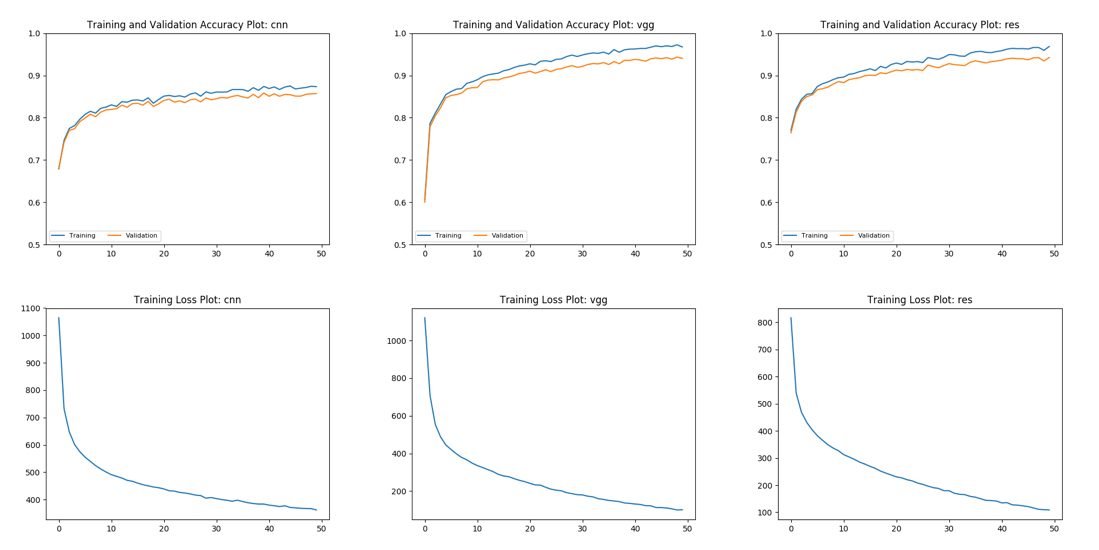


Fig 5. Accuracy on AG news dataset

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Character CNN | VGG-Type CNN | ResNet-Type CNN | **Zhang et al.** |
| Validation Accuracy | 0.857 | 0.940 | 0.942 | - |
| Test Accuracy | 0.776 | 0.838 | 0.833 | 0.8435 |

Table 3. Accuracy on AG news Dataset

**Summary**

Through the performance of the proposed methods we shall see, character level convolution neural networks and its variants outperform word-level recurrent convolution neural network, in aspects of both accuracy and performance. Also, by introducing small kernel, batch normalization, and skip layer connection, the modified architectures achieve significant improvement with similar training and inference efficiency.

1. **Discussion**

Though there exists significant difference between image classification task and text categorization task, we can still see that, with appropriate representation, some general network architectures and principles could seamlessly adapt to the text categorization task. For the variants of character level convolutional neural networks, through replacing the large kernel conv layer with small kernel conv layer, we reduce the number of parameters, but remain the same receptive field and introduce more non-linearity. Through introducing batch normalization layer after convolution layers, and skip layer connection as residual learning, we alleviate the gradient vanishing problem that the network could adapt to datasets of different scale without tuning the network. Also, through the similar performance of word level based method and character based method, deep neural networks again show the ability to capture and learn the representation.

There exists several limitations even if the performance on these tasks seems promising. The language here is limited to English, and it would be interesting to see if the character level representation and convolutional neural networks work for other languages. Also, we shall treat the convolution layers as feature extraction and it may be feasible to adapt pre-trained weights from other task to see if works.

Through the implementation of the project, we have the chance to apply deep learning to more tasks other than vision. Also, since we build the whole pipeline from ground, we learn a lot about how to design the learning system and manipulate different components to work.

**Summary**

We developed the pipeline for text categorization, and implemented the word level recurrent neural networks and character level convolution neural networks as baseline method. We also introduced several variants inspired by well-performed convolution networks architecture for vision tasks. We evaluate the proposed methods’ performance on dataset of different scales and types, and it shows that our introduced variants have gained significant improvements than the baseline methods, in both performance and efficiency.

**Reference**

[1] Lai, S., Xu, L., Liu, K. and Zhao, J., 2015, January. Recurrent Convolutional Neural Networks for Text Classification. In AAAI (Vol. 333, pp. 2267-2273).

[2] https://nlp.stanford.edu/projects/glove/

[3] Zhang, X., Zhao, J. and LeCun, Y., 2015. Character-level convolutional networks for text classification. In Advances in neural information processing systems (pp. 649-657).

[4] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[5] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

**Appendix:**

**Hyperparameters**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | RCNN | Character CNN | VGG-Type CNN | ResNet-Type CNN |
| Learning Rate | 0.01 | 0.05 | 0.05 | 0.05 |
| Momentum | 0.9 | 0.9 | 0.9 | 0.9 |
| Epoch | 50 | 50 | 50 | 50 |

Table 4. Hyperparameters for each architecture