Deep CNN-LSTM with Combined Kernels from Multiple Branches for IMDb Review Sentiment Analysis

AUTHOR: ALEC YENTER, ABHISHEK VERMA

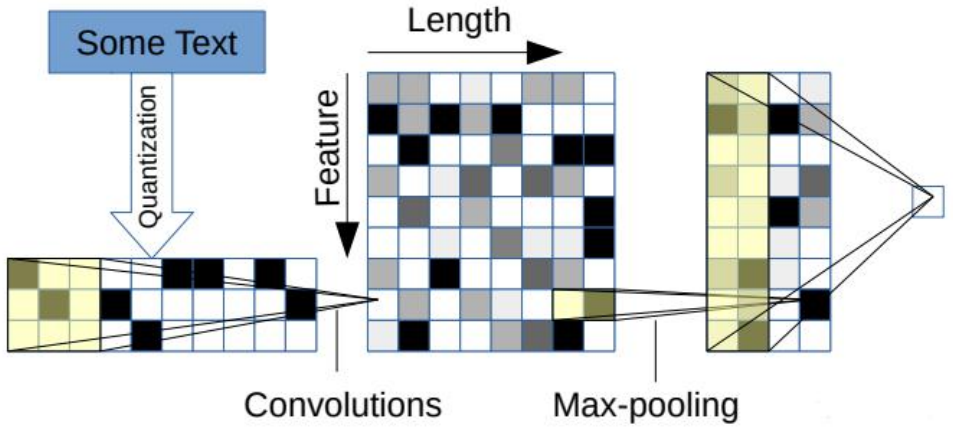
PRESENTER: JASON YAO

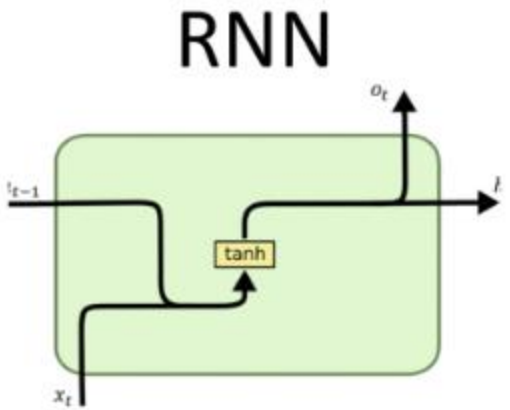
Motivation

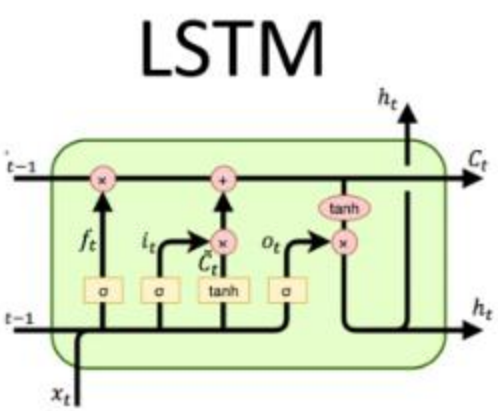
- The success of neural networks can be directed towards improvements in textual sentiment classification.
- GoogLeNet Inception model use multiple convolutional branches
- CNN learns good representations automatically, which helps get a understanding of general view of sentence
- RNNs are particularly benefitial to data that is sequential or that can value a contextual view

Background

- Main networks and layers in the paper
  - Fully-Connected (Dense) Layers: every neuron from the previous layer is connected to every neuron in the dense layer
  - Convolutional Neural Networks (CNN): usually include Convolution Layer and Pooling Layer
    - Convolution Layer: a sliding window over the input matrix with repetition
    - Pooling Layer: a sliding window over the input matrix without repetition
  - Long Short-Term Memory (LSTM): a form of RNNs where newer information in the neurons is more critical than older information
    - Recurrent Neural Network (RNN): RNN neurons have a connection to the previous neuron state in addition to the layer inputs.







Background

- Neural Network Text Classification
  - Text classification or Text Categorization is the activity of labeling natural language texts with relevant categories from a predefined set
    - Character-Level Text Classification: treat the sentence as sequence of characters, each character has a feature vector
    - Word-Level Text Classification: treat the sentence as sequence of words, each word has a feature vector
  - Sentiment Analysis: a type of text classification, classify a text into positive, negative or neural

Background

- Related work
  - in [1] a convolutional layer will look at n words at a time. Therefore, the network finds context from the n number of words nearby the word
  - In [2] the data was preprocessed before being vectorized and fed into various configuration of machine learning algorithms. The best configuration “Unigram + Bigram + Trigram” reaching the maximum accuracy of 88.94%
  - Authors in [3] use IMDb review data with a new LSTM neural network to classify sentiment of the review. The proposed LSTM layer is a biologically-inspired additive version of a traditional LSTM that produced higher loss stability, but lower accuracy. The best accuracy achieved between both LSTM models was still under 85%.

Dataset Description

- Label: 0 or 1 to represent negative and positive sentiment, linearly mapped from the IMDb's star rating system where review is rated from 1 to 10

- Reviews
  - 100,000 textual reviews
  - half for testing, the other half is further split as Fig 3
  - 234.76 words per review on average, with a deviation of 172.91 words

	Positive	Negative
Training	12,500	12,500
Validation	12,500	12,500

Dataset Description

- Review Example

Positive Example:

Although this was obviously a low-budget production, the performances and the songs in this movie are worth seeing. One of Walken's few musical roles to date. (he is a marvelous dancer and singer and he demonstrates his acrobatic skills as well - watch for the cartwheel!) Also starring Jason Connery. A great children's story and very likable characters.

Negative Example:

Not only is it a disgustingly made low-budget bad-acted movie, but the plot itself is just STUPID!!!
A mystic man that eats women? (And by the looks, not virgin ones)
Ridiculous!!! If you´ve got nothing better to do (like sleeping) you should watch this. Yeah right.

Preprocessing

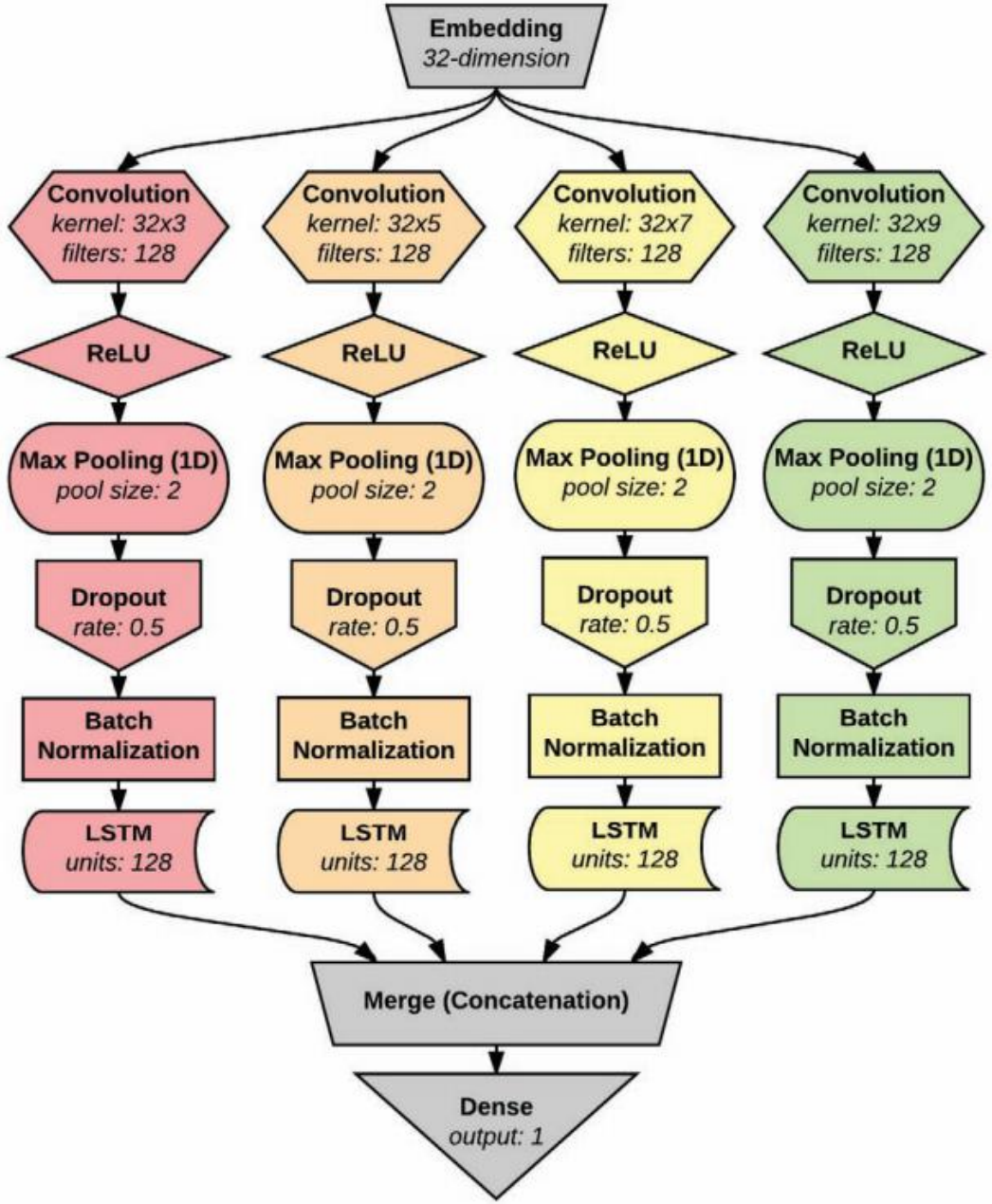
- Encode reviews encodes them into a sequence of word indices according to a dictionary of the 5000 most frequently used words in the dataset. The indices are ordered by frequency.

- Index 0 is reserved for unknown words that are not in the dictionary

- Maximum padded sequence length is 500. Longer reviews are truncated and shorter reviews are padded with zeros.

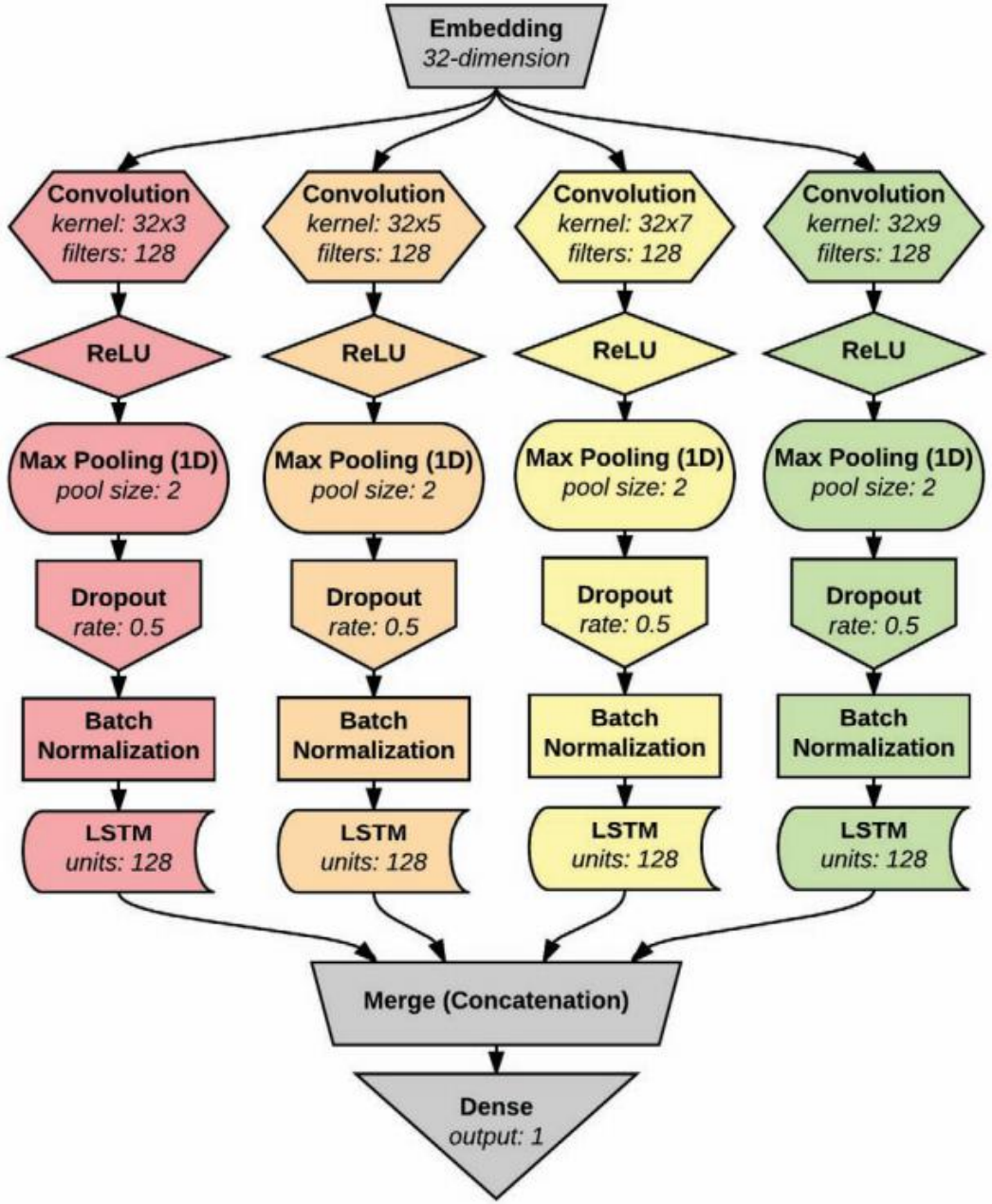
PROPOSED MODEL

- Embedding layer: A matrix of trainable weights of vectors for each word index.
  - pretrain weights word2vec and fastText were proved to have no positive effect
  - 32 best fit the dataset
- Convolution
  - kernel size c * 32 (c=3,5,7,9) which means the filter view c words at a time
  - optimal number of filters is 128, this layer's output has a shape of input height by filters (words by 128).
  - The optimal kernel regularizer was L2(0.01)



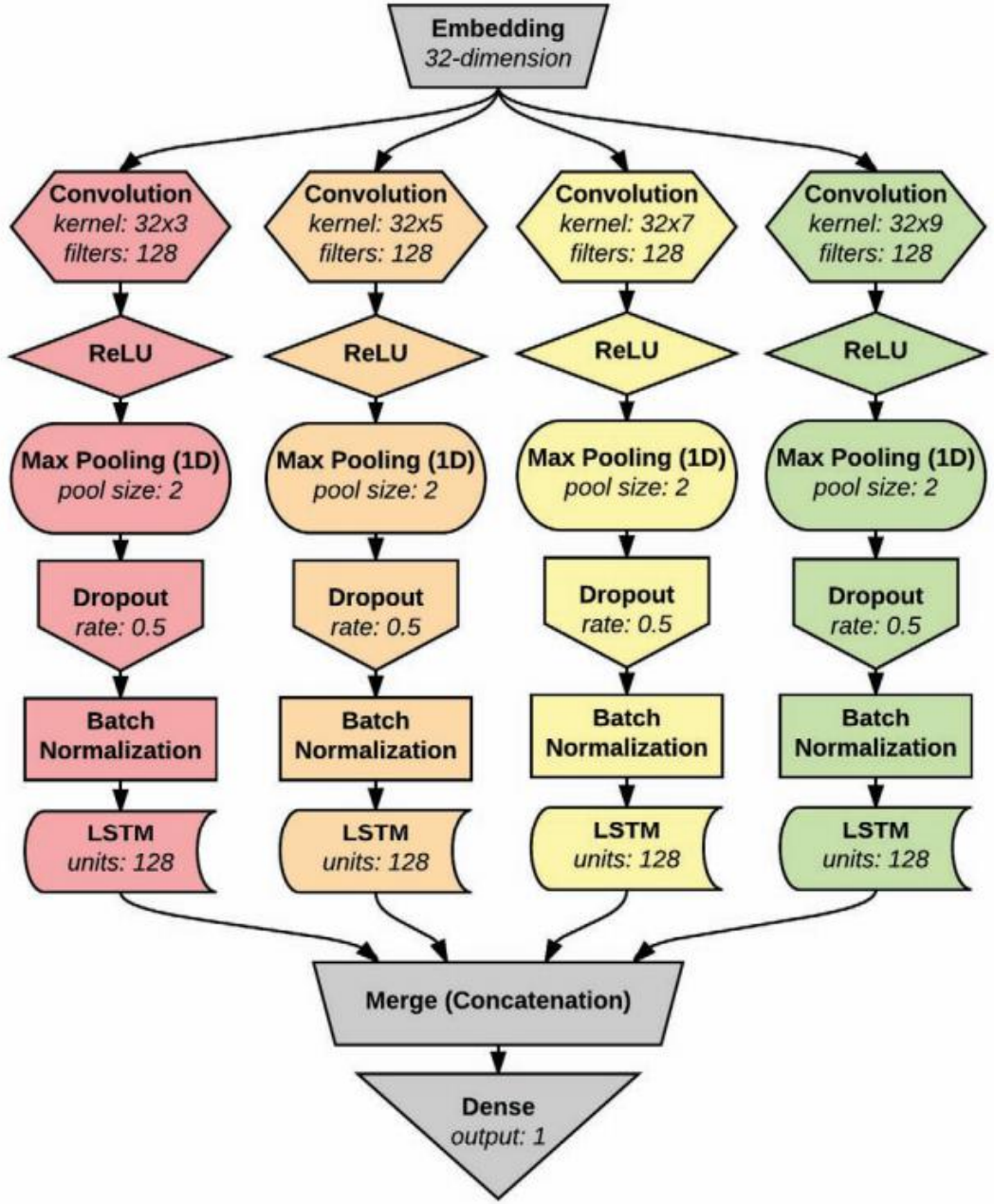
PROPOSED MODEL

- Activation: rectified linear unit (ReLU) replaces any negative outputs with zero, used in order to introduce non-linearity into the network
- 1-dimensional Max Pooling
  - optimal pool size 2
  - large pooling sizes proved to decrease accuracy
  - The result is a reduced, down-sampled version of the input.
  - reduce overfitting, while allowing for further processing



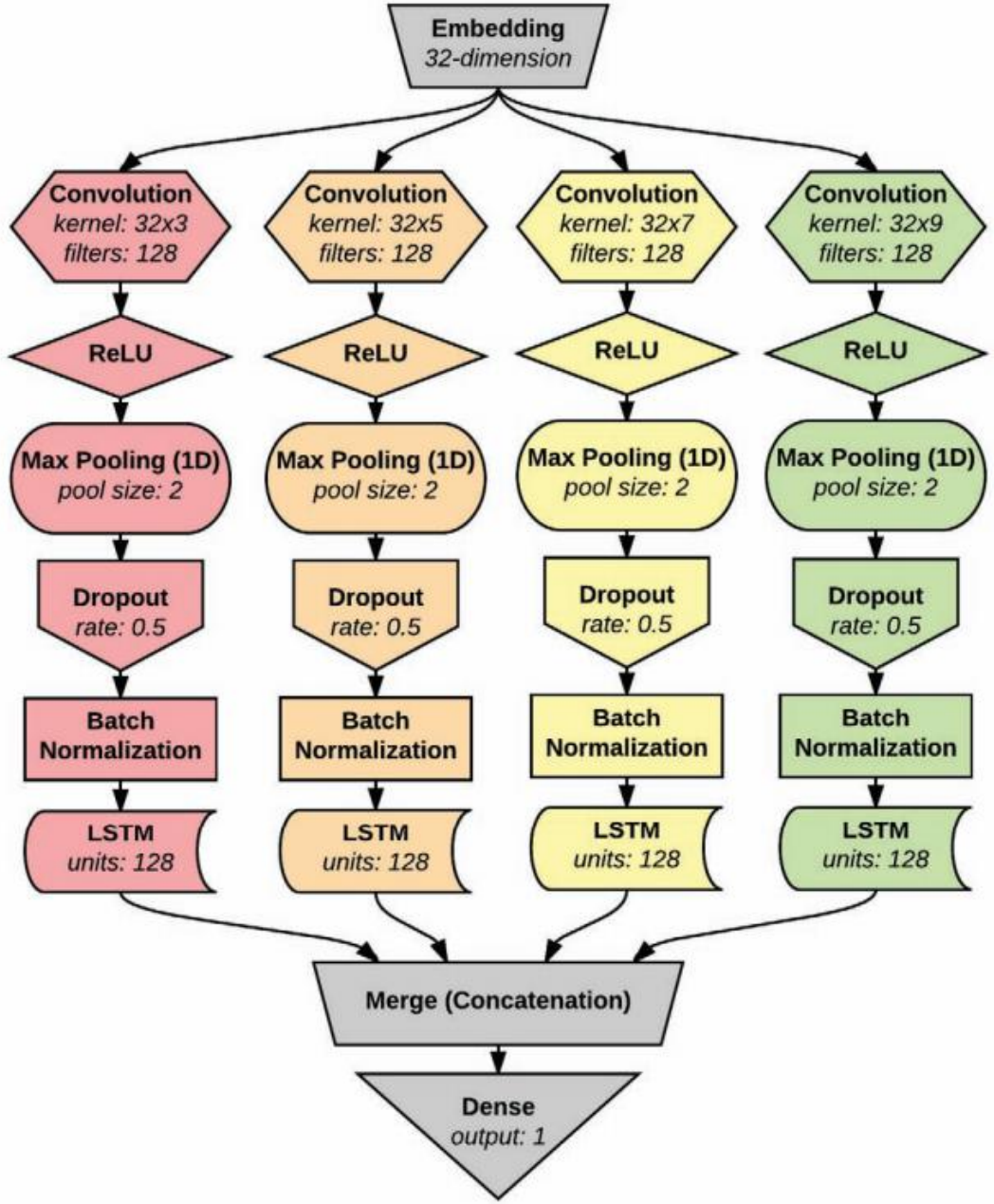
PROPOSED MODEL

- Dropout
  - randomly sets a portion of the inputs to 0
  - applied to specified 0.5 fraction of the inputs
  - found to be most helpful after max pooling and before batch normalization or, in some cases, after
  - concatenation of all the layers
  - serves to prevent overfitting and generalize the network to not focus on specific pieces of input
- Batch Normalization
  - normalizes the distribution for each batch
  - reduce internal covariate shift and therefore lead to convergence at a faster rate



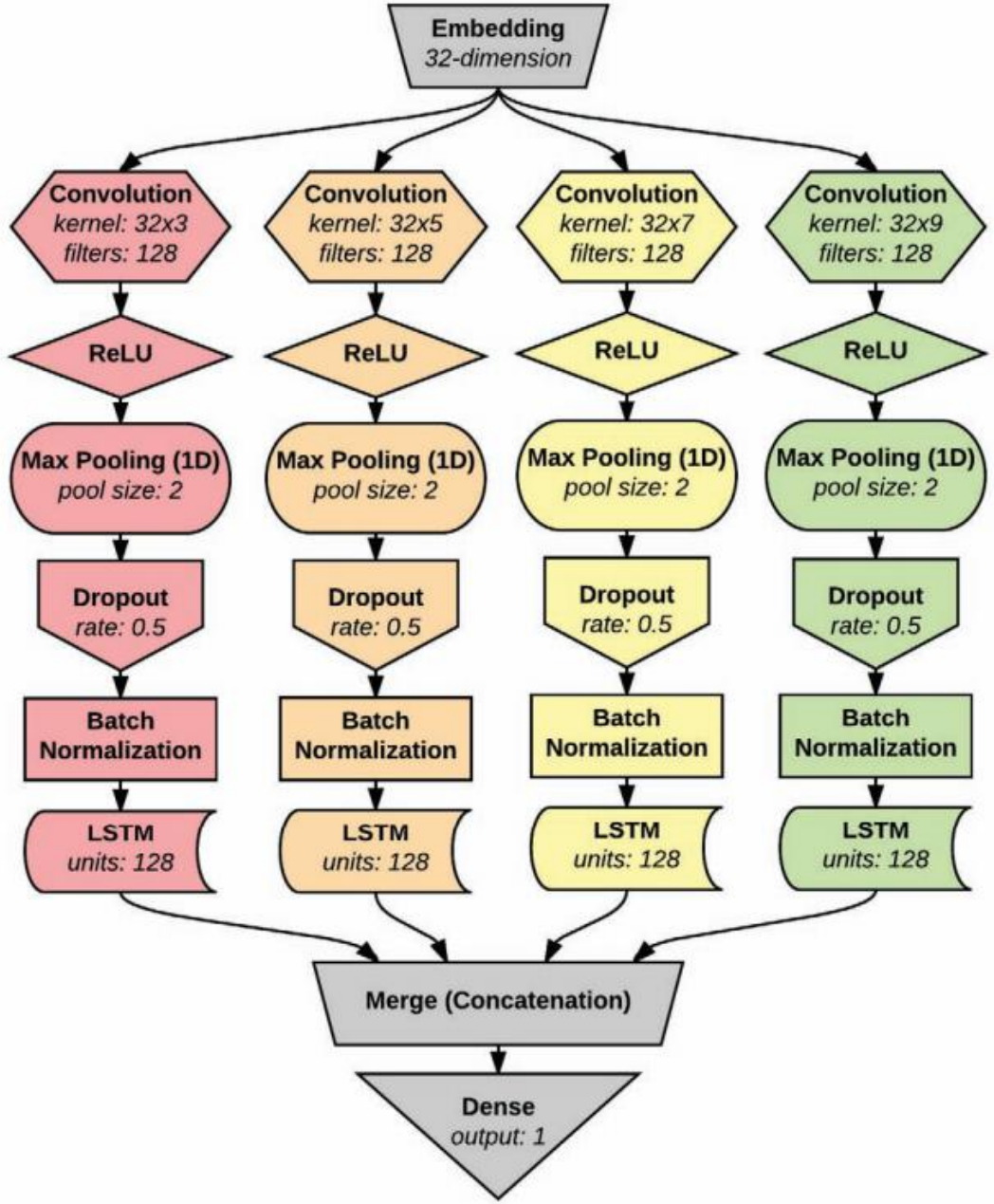
PROPOSED MODEL

- LSTM
  - 128 units
  - allows knowledge of previous input (convoluted word combinations) to influence subsequent input
- Dense
  - a fully-connected layer from the concatenated input to a single output
  - followed by a simple sigmoid activation function to conform the output between 0 and 1



PROPOSED MODEL

- Loss Function and Optimizer
  - a binary cross entropy loss function
  - Adam, RMSprop, and Stochastic Gradient Descent (SGD) were used and RMSprop found to be the best



EXPERIMENTAL SETUP

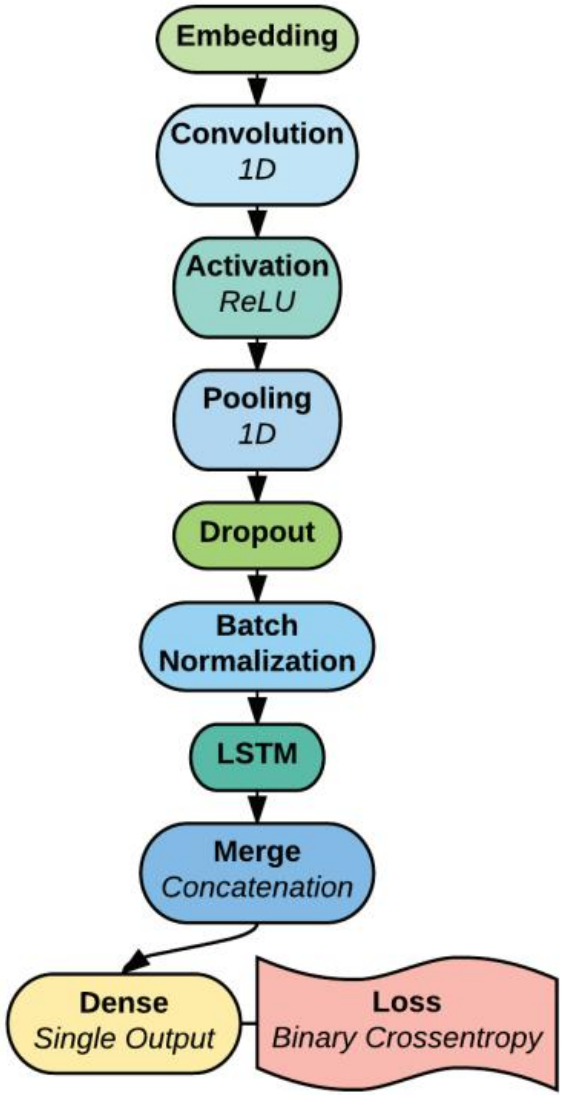
- System

The hardware used for experimenting different networks was a dual Intel Xeon E5-2690 v3 2.60GHz processors (24 physical cores / 48 logical cores) with a single NVIDIA GeForce GTX TITAN X GPU with 12 GB of VRAM and a 256 GB RAM. The machine ran Keras 2.0.4 [26] on Ubuntu 14.04.5 using TensorFlow 1.1.0 [27] backend on the GPU.

- 5000 words dictionary, 500 word length zero-padded sequence for each review
- The learning rate of optimizer was increased to 0.01 and the learning rate decay was set to 0.1

EXPERIMENTAL SETUP

- Baseline: model in paper [1]



RESULTS AND ANALYSIS

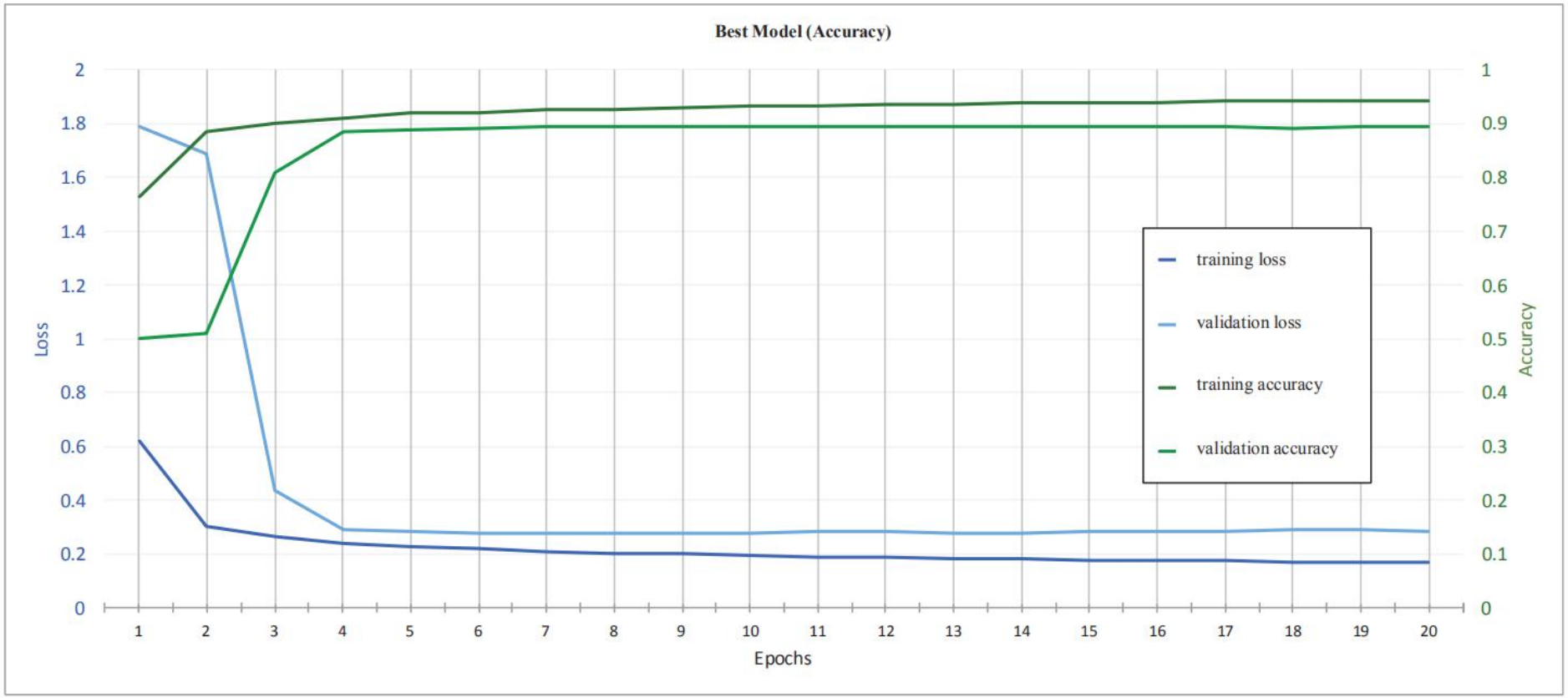


Figure 5. Graph of accuracy of top performing proposed model (Model_63).

RESULTS AND ANALYSIS

- Comparision

Table 1. Table comparing the structure, parameters, and highest accuracy of the top 5 performing proposed models in the experiments, as well as the base model.

Proposed Models	Convolution			Activation	Max Pooling	Branch Dropout	Batch Normalization	LSTM	Merge Dropout	Optimizer			Accuracy
	Branches / Kernel Sizes	*Filters*	*Kernel Regulaizer*	*Type*	*Pool Size*	*Rate*	*Present*	*Units*	*Rate*	*Type*	*Learning Rate*	*Learning Rate Decay*	*Maximum*
Model_08	3/4/5	32	None	ReLU	2	0	yes	100	0	Adam	0.001	0	0.8922
Model_16	3/4/5	32	L2(0.01)	ReLU	2	0.5	yes	100	0	Adam	0.001	0	0.8934
Model_43	2/3/4/5/6/7	128	L2(0.01)	ReLU	2	0	yes	128	0.8	Adam	0.001	0	0.8914
Model_57	3/5/7/9	128	L2(0.01)	ReLU	2	0.5	yes	128	0	RMSprop	0.001	0	0.8936
Model_63	3/5/7/9	128	L2(0.01)	ReLU	2	0.5	yes	128	0	RMSprop	0.01	0.1	**0.895**
Base Model	5	64	None	ReLU	4	0.25	no	70	0	Adam	0.001	0	0.8498

RESULTS AND ANALYSIS

- Overfitting

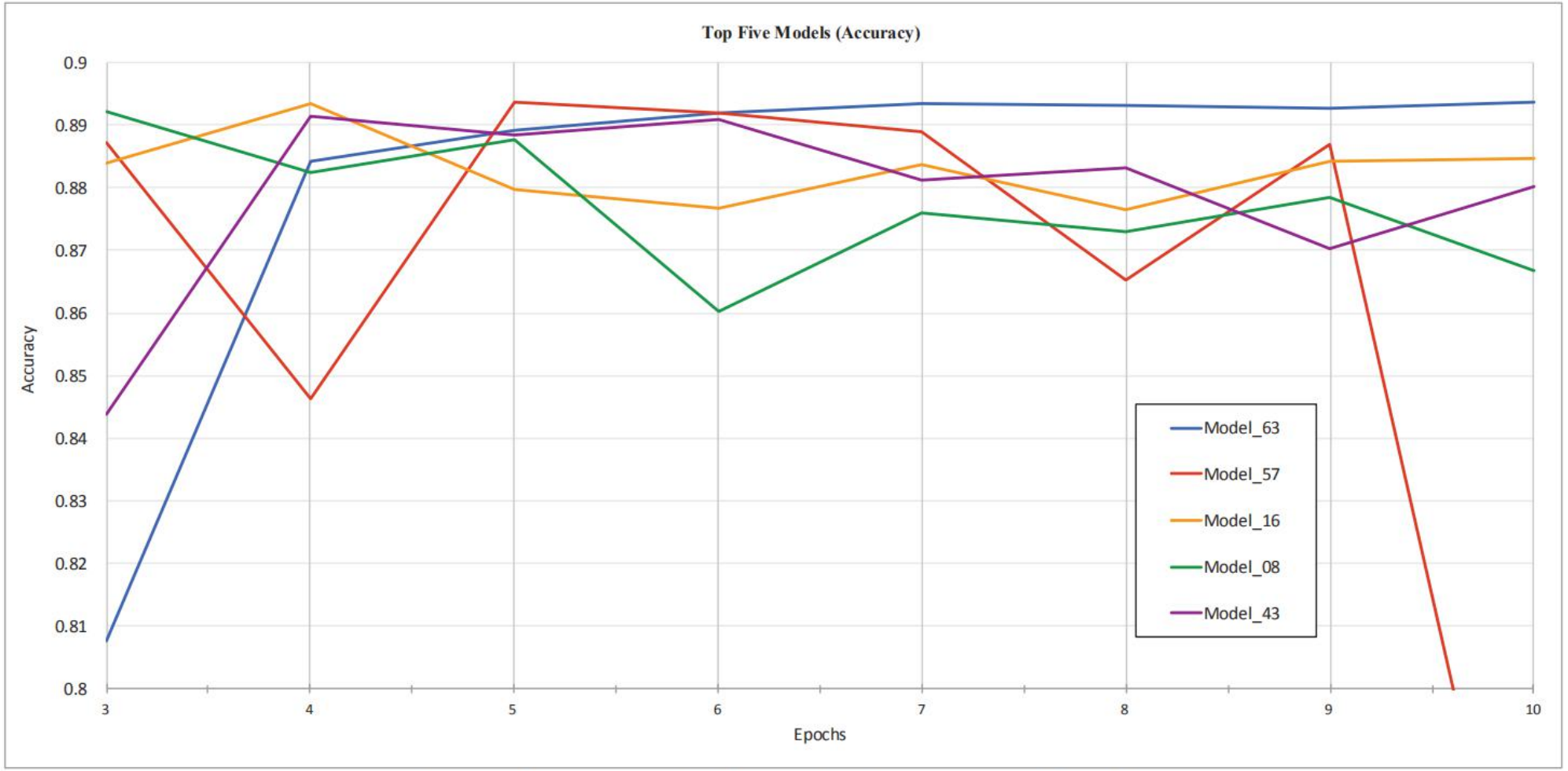


Figure 6. Accuracy graph of the top 5 performing proposed models.

Conclusion and Discussion

- This paper shows the effectiveness of the novel combined kernel from multi-branch convolution with LSTM network on the IMDb review sentiment dataset
- While the embedding was used, other methods, such as bag of words and TF-IDF, could be mixed in to produce different results
- Though word2vec and FastText no work, other pretrained weights may have better effect

Discussion

- This paper claim without learning rate decay there will be overfitting, but model_16 from figure 6 seems to no suffer from this problem
- model_16 and the optimal model_63 have quite different parameter settings but the performance are similar
- No explaination about some interesting accuracy change, for example, the big accuracy increase of model_63 from epoch 3 to 4 may reveal some information
- Use embedding size in the convolutional layer to prevent filter partial widths and cut words into pieces, no explaination or reasoning
- Didn't show the effectiveness of cnn branches