Abdoul-Nourou Yigo

Dr. Lopamudra Mukherjee

Machine Learning

December 14 2018

**Final Report**

**Text Classification with Convolutional Neural Network (CNN)**

**Convolutional Neural Network Architectures Used**

CNN is one of the variety of neural network with different layers that requires some special modeling. We have used a special model that were well-suited for our implementation. ( Zhang et al., 2016) came out with a CNN architecture we have used in our implementation . they have given precise information on how the implementation should be done. These are the procedures that should be used for the implementation. They used a tokenization procedure to tokenized the sentences which are then converted into a sentence matrix. Therefore, the row of the matrix would be the representation of each token.They have used input word vector representation from an unsupervised neural language model. One of the popular word vector is GloVe( Global Vectors for Word Representation ).

Also, for a more complex architecture they included filter region sizes that would give in depth model that would add more layers. The filter size they have recommended is ( 3, 4, 5 ). After that, a pooling strategy is used to apply to each feature map to provide a fixed length vector. With this representation, the 1-max pooling is used to extract the scalar from each feature map that would be concatenated with a fixed length. Then, this structure would be added to another procedure that is the softmax.

The softmax is the last procedure in the network. According to Wikipedia, the softmax function is a normalized exponential function. Also, it is a generalization of the logistic function. It takes an k-dimensional vector of real numbers and transforms it into a vector of real number in range ( 0, 1 ) which could add up to 1.

**Convolutional Neural Network Structural Approaches**

Now knowing the architectural structure of the CNN it is the matter of using the right CNN structure for our implementation. In (Kim, 2014) they have given different structure of the CNN that could be used in according to the size of the dataset. We have chosen to CNN-static and CNN-non-static for our implementations. in (Kim, 2014) the CNN-static structure could be implemented using a pre-trained vector from GloVe. The set of words are randomly initialized that allows the parameter to be learned accordingly. CNN-non-static is structure as the CNN-static, but the pre-trained vectors are fine-tuned for each task. In other words, in this model, a filter size is applied to different region sizes to improve the performance. We have used both models in our implementations to be able to make some comparisons in terms of efficiency.

**Implementation Procedures**

In our implementation we have used three datasets from different categories to test the performance of each model. The list of dataset we have used are the followings:

* 20newsgroups with 20 different categories grouped in 6 subcategories
  + comp.graphics, comp.os.ms-windows.misc, comp.sys.ibm.pc.hardware, com.sys.mac.hardware, comp.windows.x
  + rec.autos, rec.motorcycles, rec.sports.base, rec.baseball, rec.sport.hockey
  + sci.crypt, sci.electronics, sci.med, sci.space
  + Misc.forsale
  + talk.politics.misc, talk.politics.guns, talk.politics.mideast
  + talk.religion.misc, alt.atheism, soc.religion.christian
* bbc news dataset with 5 categories
  + business
  + entertainment
  + politics
  + sport
  + tech
* bbc sport one category
  + Sport
    - 5 Subcategories
      * Athletics
      * Cricket
      * football
      * rugby
      * Tennis

With these datasets we have done a total of six different evaluations using CNN-static and CNN-non-static for each dataset.

**Important Resource libraries used for the implementation**

**Keras**

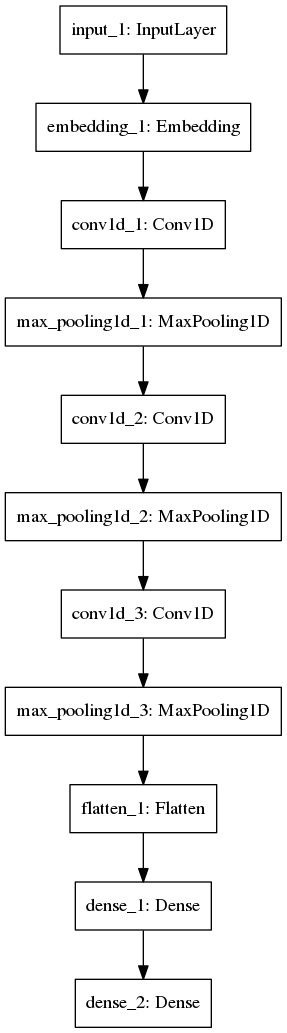
We used keras to build the CNN models. The layers used are:

* Concatenate: this layer concatenates a list of inputs
* Conv1D: a tensor that returns a result of 1D convolution
* MaxPooling1D: Max pooling for temporal data
* Flatten: it returns a tensor and can reshape the tensor into one-dimensional convolution
* Dense: In a dense, all nodes in the previous layer connect to the nodes in the current layer

With the above layers we have the following models:

**CNN-static**

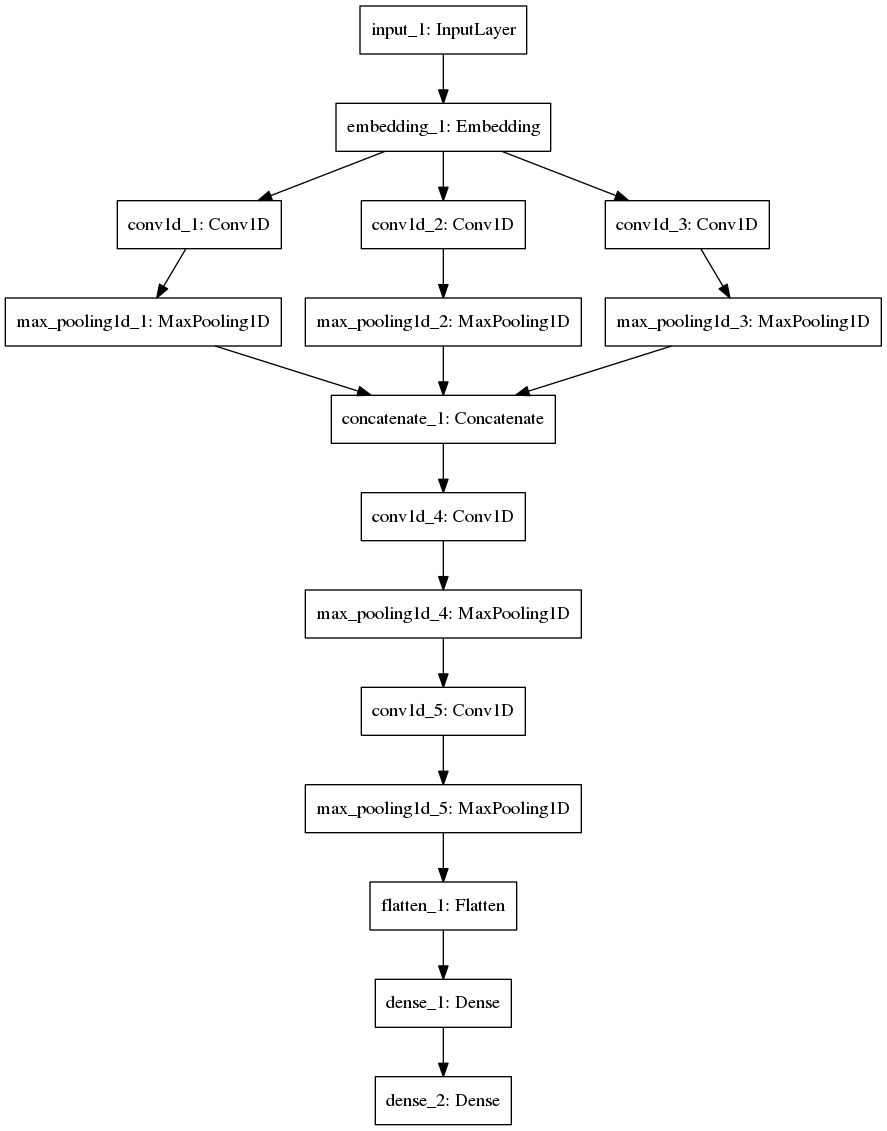
This is the CNN-static model structured by Keras



**CNN-non-static**

This is the CNN-non-static structured by keras. We can see the embedding is splitted in three different regions. As said previously, we have used the filter size

(3,4, 5). The embedding layer splits the the input in 100 map features.



**Other Libraries used**

**TensorFlow**

This is an open source library is used to handle very high computation since we are dealing with considerable amount data.

**Numpy**

This library is used to structure data.

**Matplotlib**

This library is used to plot the results.

**Sklearn**

This library is used to download the data.

**Evaluation of CNN-static and CNN-non-static**

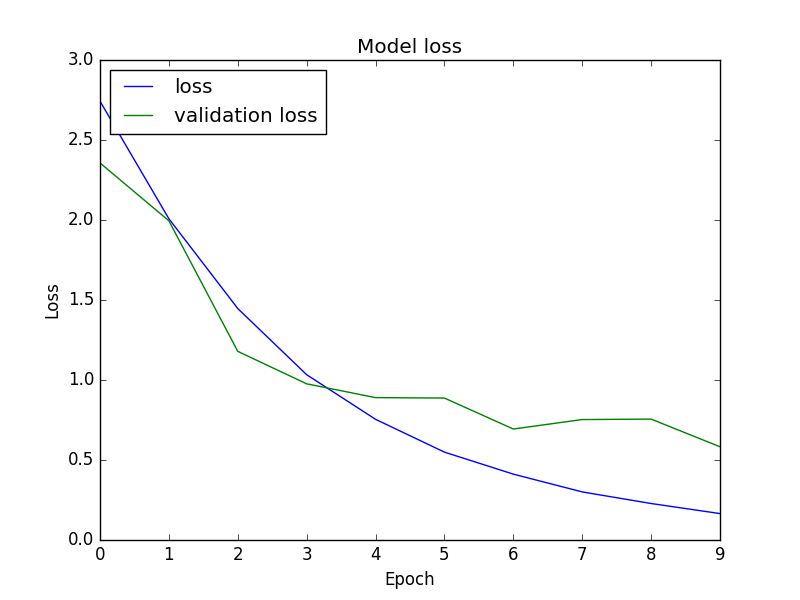
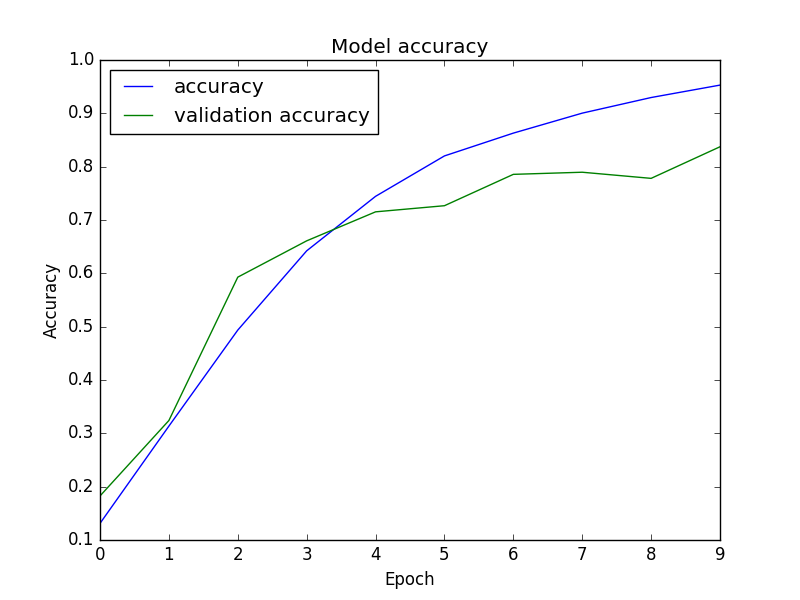
After the tests, we have found valuable results with the specific dataset we have used.

**Evaluation of the 20newsgroups dataset**

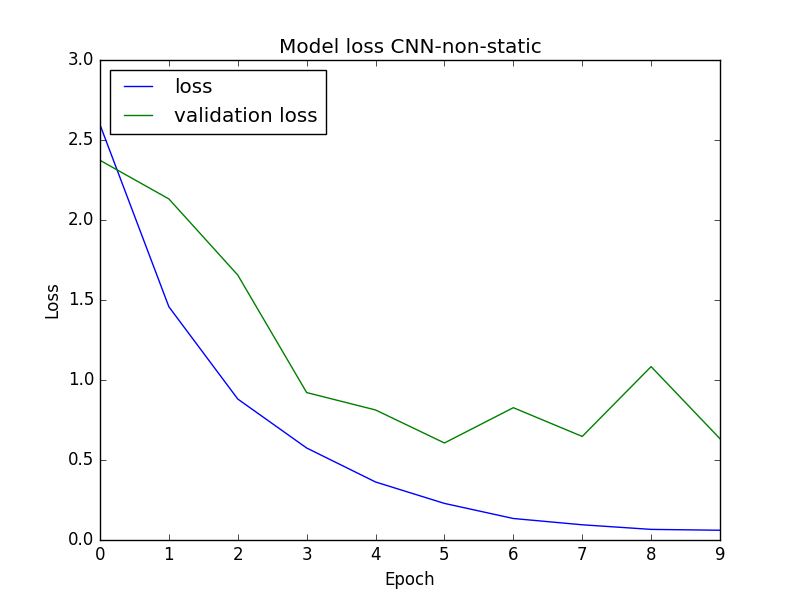
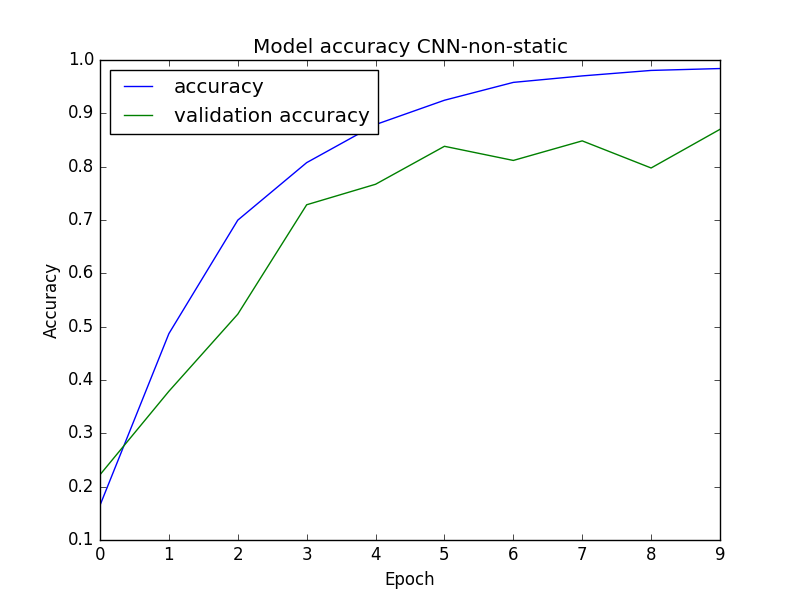
**Training samples**: 9052 samples

**Validation Samples**: 2262 samples

**CNN-static**



**CNN-non-static**



The above figures are showing the following results:

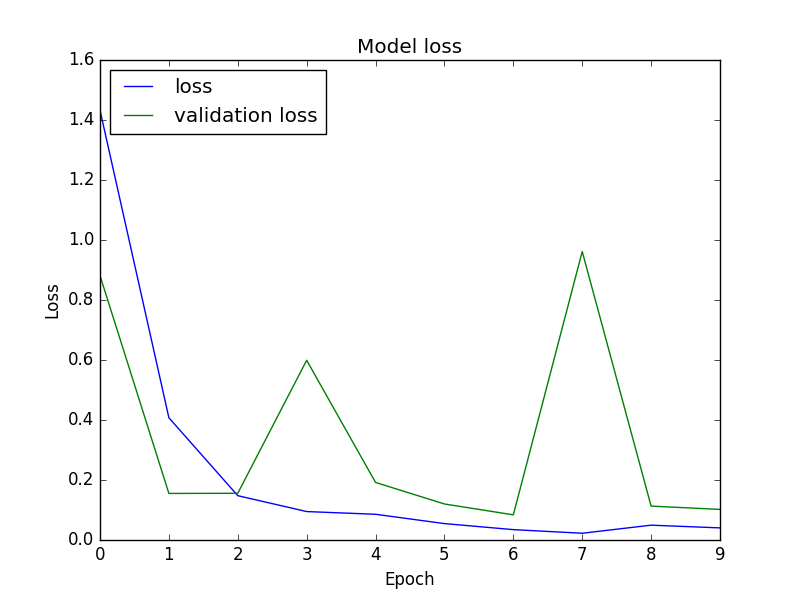
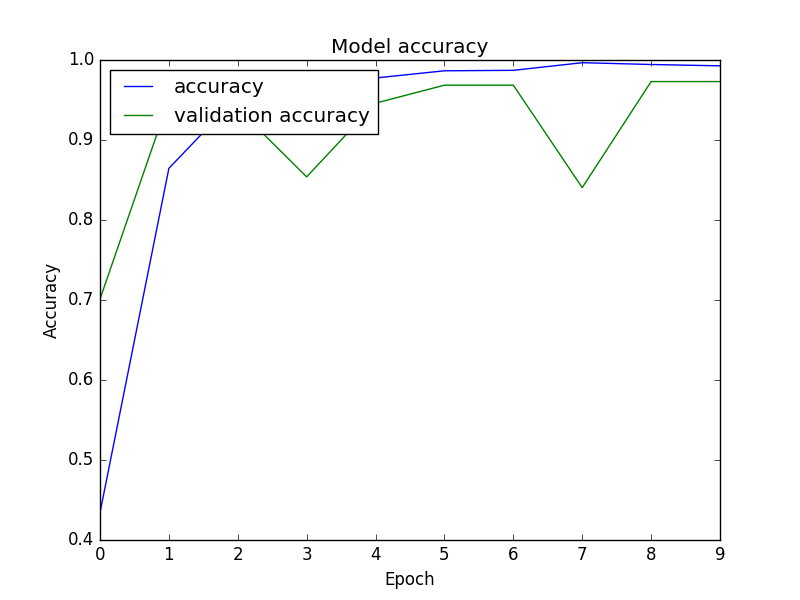
|  |  |  |
| --- | --- | --- |
|  | CNN-static | CNN-non-static |
| 20newsgroups | Accuracy: 0.9529  Loss: 0.1652  Validation accuracy: 0.8373  Validation loss: 0.5828 | Accuracy: 0.9840  Loss: 0.0610  Validation accuracy: 0.87  Validation loss: 0.6328 |

**Evaluation bbc news Dataset**

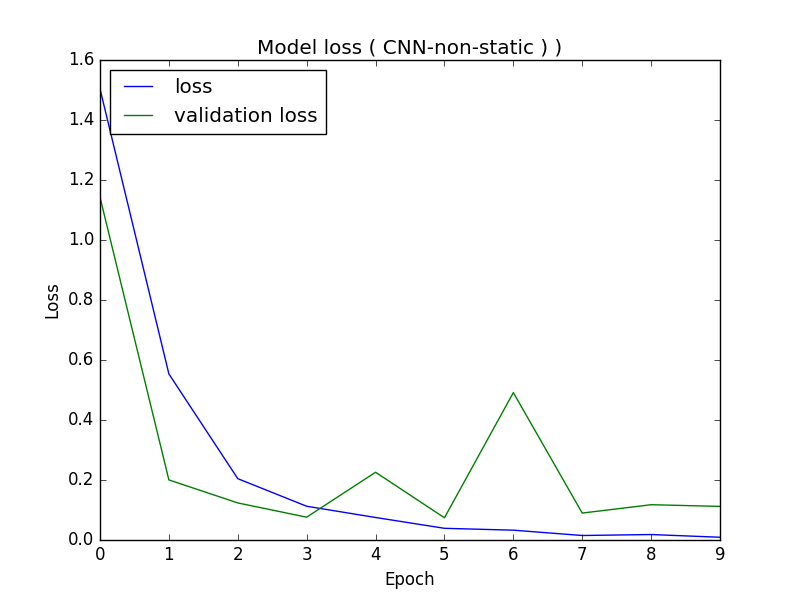
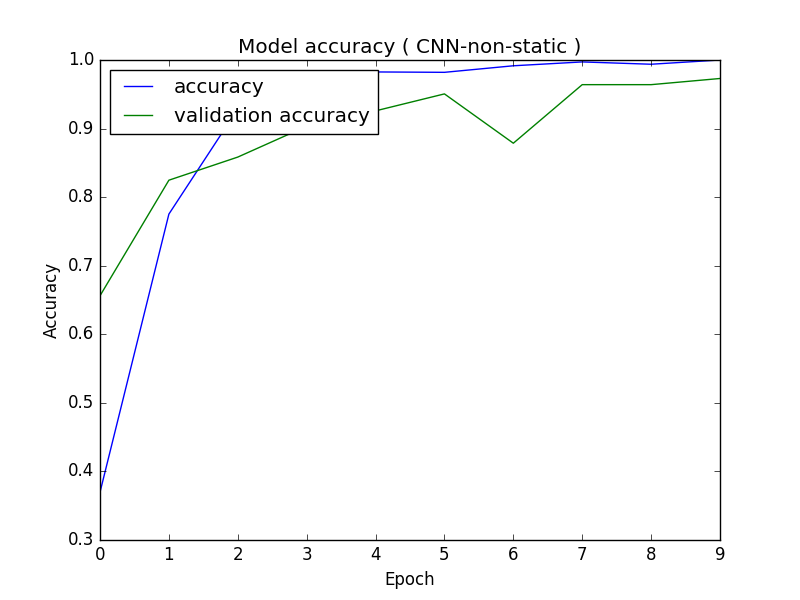
**Training Samples**: 1780 samples

**Validation Samples:** 445 samples

**CNN-static**



**CNN-non-static**



The above figures are showing the following results:

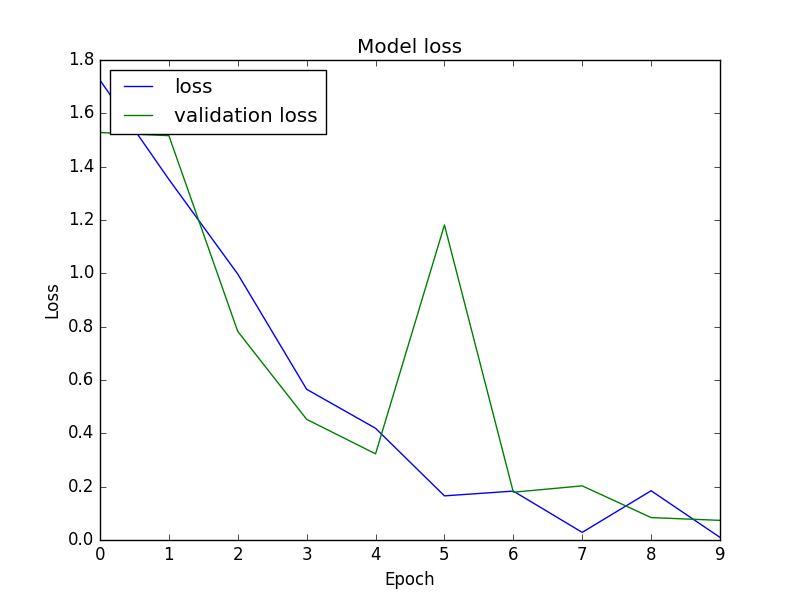
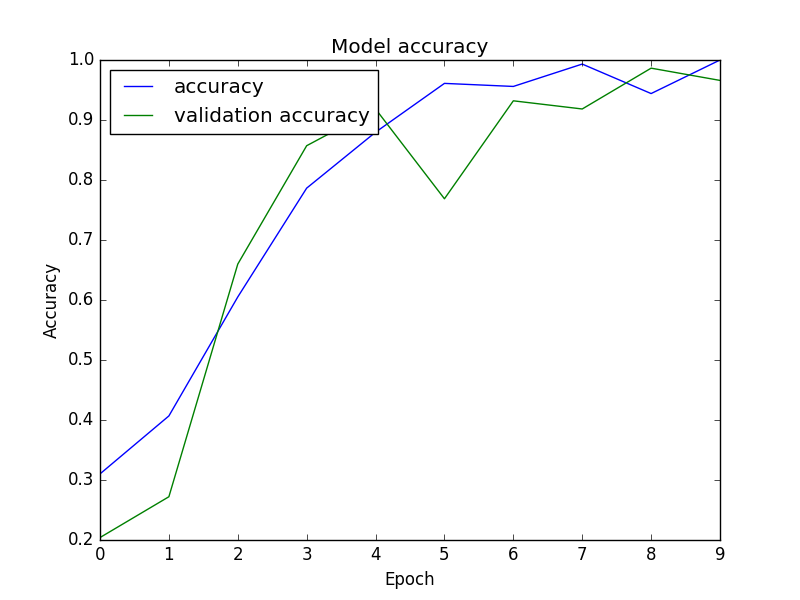
|  |  |  |
| --- | --- | --- |
|  | CNN-static | CNN-non-static |
| Bbc news | Accuracy: 0.9508  Loss: 0.2971  Validation accuracy: 0.9048  Validation loss: 0.1998 | Accuracy: 0.9983  Loss: 0.0091  Validation accuracy: 0.9798  Validation loss: 0.1120 |

**Evaluation of bbc Sport**

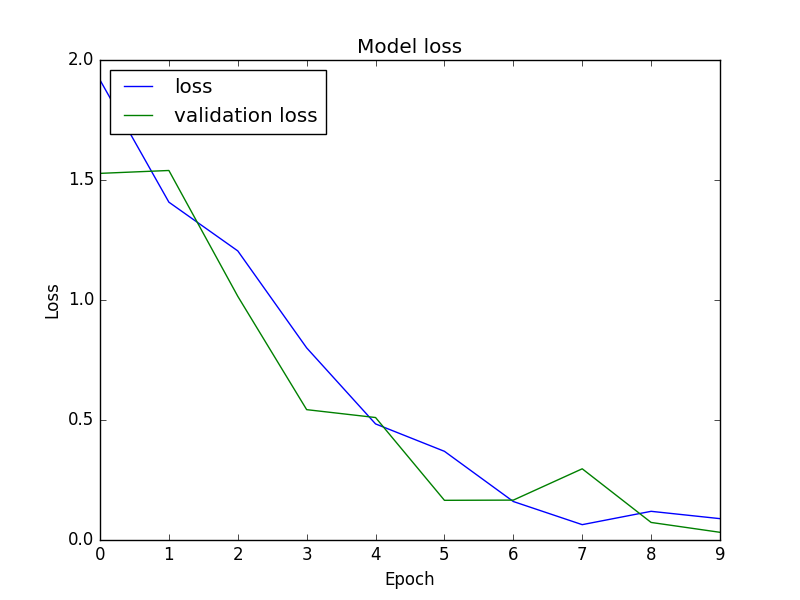
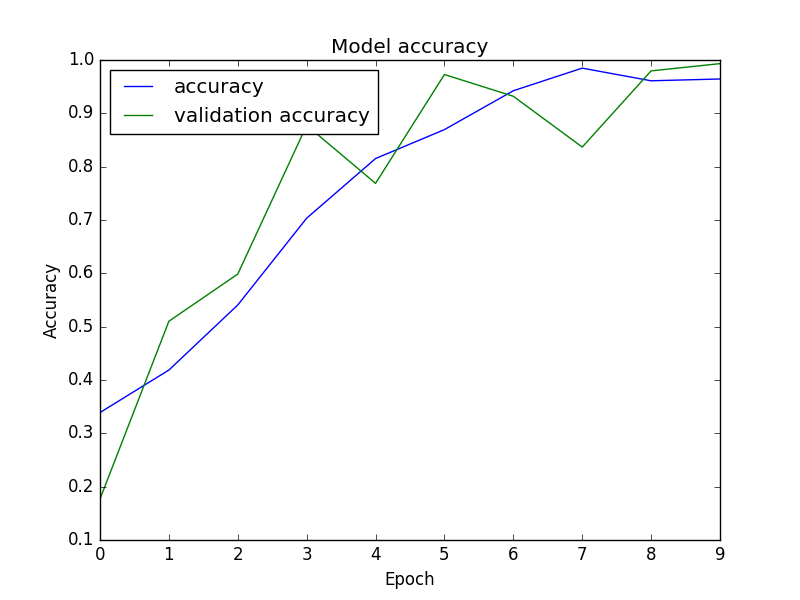
**Training Samples**: 590 Samples

**Validation Samples:** 147 samples

**CNN-static**



**CNN-non-static**



The above figures are showing the following results:

|  |  |  |
| --- | --- | --- |
|  | CNN-static | CNN-non-static |
| Bbc sport | Accuracy: 0.9780  Loss: 0.1192  Validation accuracy: 0.9524  Validation loss: 0.1472 | Accuracy: 0.9356  Loss: 0.3550  Validation accuracy: 0.9796  Validation loss: 0.0638 |

With all the results we have had, we can see that the CNN structures are performing well on all the datasets. For the figures we have two curves because we have splitted in two sets. The training dataset represents 80 percent of the dataset and the validation set represents 20 percent of the dataset. Therefore, the accuracy and loss would be referring to the training set. The validation accuracy and validation loss would be referring to the validation dataset.

The validation accuracy and validation loss are used to double checked the efficient of the network to make sure that the network does not overfit or underfit. If the training loss is less than the validation loss set, the network might overfit. In our results, we did not experience this kind of situation. Therefore, the system have performed considerably well during the tests.

We have tried to use crawled data to do another evaluation, but the results we got were not good enough to be documented. Therefore, since, there were a kind of time issues, we have tried to leave this portion for feature research and implementation.

Comparing the CNN-static and CNN-non-static, we could observe that the CNN-non-static has performed well with the large datasets ( 20newsgroups, bbcnews ) with the accuracy of 0.9840 and 0.9983 respectively. However, the CNN-static bypasses the CNN-non-static with small size of dataset( bbcsport ) with the accuracy of 0.9780 against 0.9356.

**Summary of the Evaluation**

This is a summary of all the results from different evaluations. We have done our evaluations in a decreasing manner. In other word, we started large dataset with 20 categories. A medium dataset wit 5 categories. Finally, a dataset with 1 category.

|  |  |  |
| --- | --- | --- |
|  | CNN-Static | CNN-non-static |
| 20newsgroups with 20 Categories | 0.9529 | 0.9840 |
| bbc news with 5 categories | 0.9910 | 0.9983 |
| bbcsport with 1 category | 0.9780 | 0.9356 |

**Conclusion**

From our motivation, we were able to bypass some complexities to learn the concepts of convolutional neural network ( CNN ). Doing in depth research, we were able to use the results of the research to make to make some implementations with various structures the CNN. With these implementations, we were to make some evaluations on some specific datasets that have allowed us to understand the CNN performances in terms dataset sizes. The next step would be the integration of this kind of system in our application. Of courses, we would have to do a more profound research to get in depth understanding of the CNN.

**References**

[Kim2014] Yoon Kim. 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[ Zhang and Wallace2016 ] Ye Zhang and Byron C. Wallace. 2016.. Wallace. 2016. A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.

D. Greene and P. Cunningham. "Practical Solutions to the Problem of Diagonal Dominance in Kernel Document Clustering", Proc. ICML 2006.

Wikipedia contributors. (2018, December 13). Softmax function. In *Wikipedia, The Free Encyclopedia*. Retrieved 19:26, December 14, 2018, from <https://en.wikipedia.org/w/index.php?title=Softmax_function&oldid=873500022>

Keras Documentation. Keras: The Python Deep Learning library. Available at: <https://keras.io/>