Iran University of Science and Technology
Deep Learning (Fall 2019)

Memotion Analysis

Arash Moshirnia
Department of Computer Engineering
Iran University of Science
and Technology
arash_moshirnia@ce.iust.ac.ir

Arian Shariat
Department of Computer Engineering
Iran University of Science
and Technology
arian_shariat@ce.iust.ac.ir

Abstract

Since the previous decade, Memes have grown to be one of the hottest subjects on the internet and arguably, the most common kind of comedy seen on social media platforms nowadays. But despite their huge growth, there is not much attention towards meme emotion analysis. Because of this growth and the kind of effect social media has on pop culture, analyzing memes might come in handy and even become a necessity in the near future.
Memes are usually images with text written inside, so the kind of sense and humour they carry is a result of these two parts together. Therefore, to classify their kind of sentiment and humour, it is necessary to consider both parts; Which is our approach in solving this problem.

1 Introduction

The Memotion Analysis task defined on Semeval 2020 competition consists of 3 sub-tasks:

- **Task A - Sentiment Classification**: Given an Internet meme, the first task is to classify it as positive, negative or neutral meme.
- **Task B - Humor Classification**: Given an Internet meme, the system has to identify the type of humor expressed. The categories are sarcastic, humorous, and offensive meme. If a meme does not fall under any of these categories, then it is marked as a other meme. A meme can have more than one category.
- **Task C - Scales of Semantic Classes**: The third task is to quantify the extent to which a particular effect is being expressed.

Our main task in this project is the first sub-task mentioned above and our approach is based on Deep Learning methods.

1.1 Challenges

Notable challenges in this project:

- The first and most difficult challenge in this task, is the fact that determining the type of sentiment in a given meme (and the concept of humour in general) could be hard even for humans themselves, let alone machines. So it should be noted that this task might be a long way from getting solved.

(LateX template borrowed from NeurIPS 2019)

- Another challenge faced when working with memes, is the use of texts and images together. In order for networks to work with this kind of complex problems, it is required for them to be deep and complex enough. This also leads us to another challenge, which is extracting the text out of the image; but fortunately this is already done for us in the dataset available for this task.
- The last challenge of this task is that since the main work should be done on the text, in the absence of sufficient training data, Augmentation techniques can't be used to generate new instances. So other techniques should be used to make the most out of the available data.

2 Related work/Background

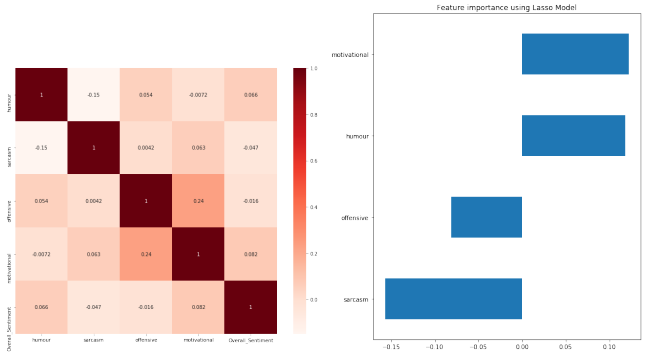
Since memes are a new trend, not much work has been done on them. The task of identifying a meme's sentiment and classifying it's humour is an unsolved problem and as of this date, no kind of related work has been published yet.

3 Proposed Method

3.1 Preprocessing

The dataset used for this task, is the one provided by the Semeval contest. It contains around 7500 instances, each having these attributes: Image name, Image URL, OCR extracted text, Corrected text, The extent of it's Humour, Offensiveness and Sarcasm in fuzzy terms, Whether it's Motivational or not, and it's Overall Sentiment .
As with any Deep Learning project, lots of preprocessing had to be done on this dataset to make it usable for training. The steps taken in preprocessing our training data is as follows:

1. At first, any instance containing no OCR extracted text and no corrected text was useless and hence, removed.
2. Due to an abnormality in the dataset, a few instances that didn't contain the Overall Sentiment value had a shifting problem; as each value was mistakenly placed in the column on it's left. So these instances were shifted to right once.
3. Then, using the *nltk* library, all the stop words (such as am, is, are) were removed.
4. Next, it was noticed that the corrected text contained noises such as redundant URLs, usernames and meaningless characters. These noises were also removed from the meme's text.
5. Then, the fuzzy terms used to describe how humorous, sarcastic, offensive and motivational the memes are were normalized to a number between 0 and 1.
6. And finally, after feature-importance engineering of the data We figured out that 2 of the 4 included categorical data (Motivational and Sarcasm) aren't as important and decisive as the other two. So we omitted them from our inputs. The related diagrams come bellow:



(a) Feature Importance Heat-map (b) Feature Importance by Lasso

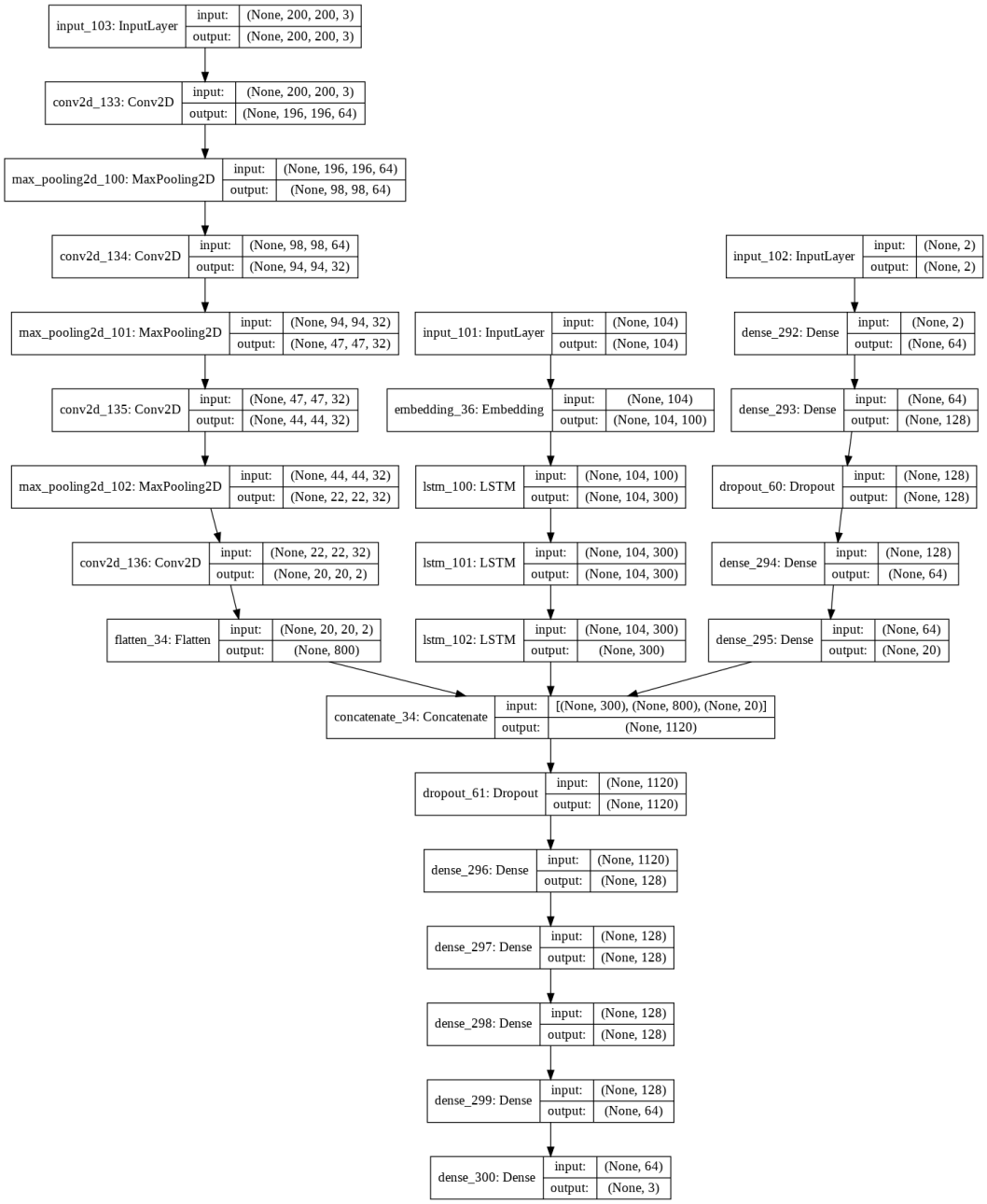


Figure 2: Network's Graph Representation

3.2 Architecture of Our Neural Network

The network we designed is implemented by Keras's functional API. Due to the format of our data, three different input layers are fed into the network:

- Meme's Image
- Meme's Text
- Meme's Categorical Data (Humour, Sarcasm, Offensiveness, Motivational)

Therefore, we have three branches in our network.

1. A Convolutional Neural Network(CNN) is used for learning the meme's image and extracting the features related to it.
2. A Recurrent Neural Network(RNN) model is used for analyzing and learning the meme's text features.
3. A Multi-Layer Perceptron(MLP) model is used for learning the categorical data.

The results of these different branches are then concatenated and fed to a final MLP which means to classify the meme's Overall Sentiment based on the information retrieved from the three sub-models. The output layer of our model is a Dense layer with 3 neurons (one for each class) and a Softmax activation.
Figure 2 is a graphical representation of our network.

4 Results

The final count of the available training data to work with (after preprocessing and downloading images) is 5968. From this amount, We kept 85% for training (5027) and the rest for validation.

4.1 Baseline

From the total number of our training data, 2971 belong to the positive , 1720 belong to the neutral and 336 belong to the negative categories. So the Baseline approach to this problem would be classifying every meme as positive; which would give us an accuracy of 59.10%.

4.2 Our First Approach

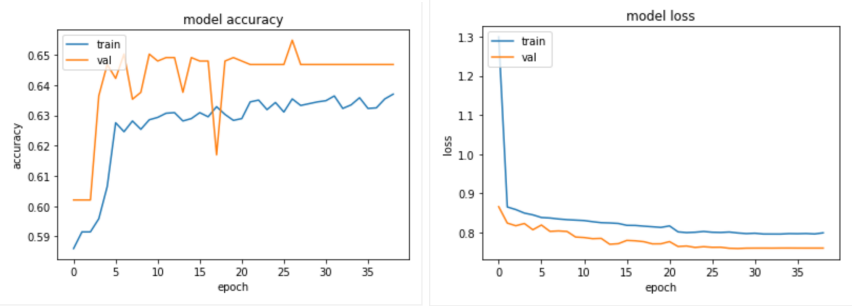
Using the network described in the Proposed Method section, We trained our model on the training data for 25 epochs. This resulted in the highest validation accuracy of 64% and the final validation accuracy of 54%; which means our model beat the baseline method but started overfitting on the data. This is also pretty obvious from the huge gap between the train and validation accuracy.

4.3 Our Final Approach

To overcome the overfitting problem we faced in our previous approach, we tried using some regularization techniques.

- First of all, since the number of instances in the negative class is much less than the other two, We computed and used class weights in order to balance the data.
- We used L2 regularizers in the CNN and RNN branches and added Dropout layers to our model
- And then, we used some features from Keras's callbacks package. For instance, we wrote a function to decrease the model's learning rate every 30 epochs. We also used Checkpoints to stop upgrading our weights after model started overfitting.

Despite all of these attempts, our model still couldn't achieve an accuracy higher than 65%. But the most important improvement over the previous approach is the ultimate reduction of the gap between the train and validation accuracy; which is a sign that our model isn't overfitting the same as before anymore and it's generalization power has improved.
This model's accuracy and loss charts is shown in figure 3.
Also the model's classification report can be viewed in figure 4. This report gives us some very useful information about how the model performed on each class and in general. For example, it can be seen that none of the inputs of the class 0 (Negatives) were classified correctly. This was predictable because of the excruciatingly small number of instances available in that class. We can also again see that our accuracy on the validation data is 66%.



(a) Final Model's Train and Validation Accuracy

(b) Feature Model's Train and Validation Loss

Figure 3: Final Model's Related Charts

	precision	recall	f1-score	support
0	0.00	0.00	0.00	68
1	0.57	0.23	0.33	283
2	0.67	0.95	0.78	562
accuracy			0.66	913
macro avg	0.41	0.39	0.37	913
weighted avg	0.59	0.66	0.59	913

Figure 4: Final Model's Classification Report

5 Discussion

As it was mentioned in the previous sections, this task is fairly complex because it's about humour; which is closely related to how human's understand. Even different people might have different opinions on humour. It happens much that something funny to a person, is offensive to another. So it's difficult for machines to learn and classify things related to it.
But it should be mentioned that in our experiment, we managed to achieve a higher accuracy than the baseline model. This means that this task can be solved using Deep Learning approaches in the presence of a good dataset; Which is the main reason our model couldn't achieve better results.
Although we managed to reduce overfitting to a great extent, since the number of inputs is very very few for a Deep Learning task, it fails to learn more useful patterns and features and hence, overfits in the end. To avoid this we should either increase the amount of our training data or use other methods and try more tools, which we couldn't do.