How Do You Feel, My Dear

Sara Manfredi[1]

Università degli Studi di Milano, Milano, Italy sara.manfredi1@studenti.unimi.it

1 Introduction

Recently, emotion detection in text has received attention in the literature on sentiment analysis. Detecting emotions is important for studying human communication in different domains, including fictional scripts for TV series and movies. The project aims at studying fictional scripts of several movies and TV series under the emotional profile of the main characters. We constructed an LSTM Recurrent Neural Network for the prediction of the VAD values and used the results to show the emotional profile of a character, how it changes over time and how is it affected by the other characters in the movie.

1.1 VAD dimensional model

This model considers affective states to be best described relative to a small number of emotional dimensions: *Valence* (corresponding to the concept of polarity), *Arousal* (degree of calmness or excitement) and *Dominance* (perceived degree of control over a situation) [2].

1.2 Related work

Xie et al. in [3] proposed a method to perform multi-dimensional sentiment score prediction, as VAD or VAI (Valence Arousal Irony), based on the assumption that the dimensions can be dependent on each other. In particular, they tested different types of Neural Networks (CNN, LSTM, a combination of CNN and LSTM and Hierarchical Attention Network) to make the predictions and each of the networks was used in two modes: internal and external. The internal mode incorporates multidimensional relations into neural networks to construct the vector representation of given sentences based on word embeddings while the external model builds a linear regression model that can capture relations between dimensions to refine the predicted results that are output by the neural networks. In both modes, they construct an encoder for each dimension and then combine the results in the two modalities.

2 Research question and methodology

The project aims to construct a model capable of predicting VAD values of a text using a LSTM Neural Network with three output variables: one for each dimension. We decided to use a dimensional approach, and not a categorical one (such as positive and negative classes or Ekman's [1] six basic emotions), for the representation of the affective states because in this way we can provide a more fine-grained (real-value) sentiment analysis. Furthermore, we decided to use the dependent assumption of the dimensional values. It is different from the approach proposed in [3], even if we used the same representation for the affective states and the same assumption on the dependence of the dimensional values, because here we construct a single encoder for the three dimensions.

2.1 Model overview

The model used for the prediction consists in:

- an Embedding Layer that transforms each word into a vector representation and so each text into a matrix
- a Dropout Layer
- an LSTM Layer
- a Dense Layer with ReLU activation function
- a Dense Layer with 3 output and linear activation function

(See Fig. 1).

For the construction of the model, the Tensorflow [1], Keras [2] and Gensim [3] toolkits were used.

2.2 Formal definition of the problem

The problem belongs to the realm of sentiment analysis of a text. The main issues with the problem are:

- Represent a text in a way that can be used in a model.
- Integrate into the representation of the text the information about the sequence of the words.
- Learn the words, named valence shifter, that are able to change the sentiment of the text.

We decided to represent the text as a matrix where each row represents a word and where the sequence of the words is preserved. Each word is represented as a vector that is taken from pre-trained vector representations of words. The use of the LSTM consents us to use this matrix as representation of a text and to automatically learn the valence shifters in a data-driven manner[Chapter 13.3] [5].

[1] `https://www.tensorflow.org/`
[2] `https://keras.io/`
[3] `https://radimrehurek.com/gensim/`

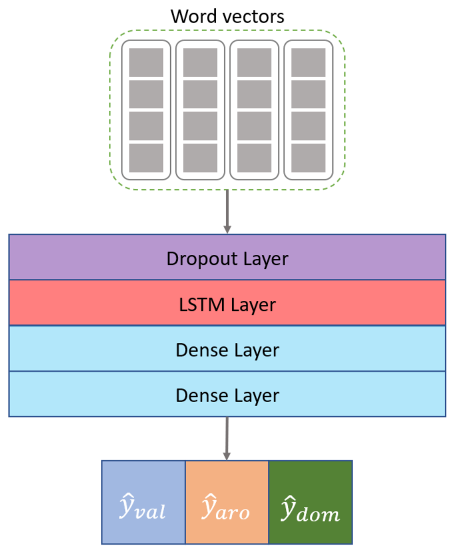

Fig. 1: Conceptual diagram of the model.

3 Experimental results

3.1 Dataset used for the creation of the model

The dataset used for training the model is EMOBANK [2], a corpus of 10k English sentences that were annotated with dimensional emotion metadata in the Valence-Arousal-Dominane (VAD) representation format. The dataset was manually annotated from both the reader's and writer's perspectives. The values of VAD are comprised between 1 and 5. We decided to use the Weighted average of reader and writer annotations. The dataset was divided in 6:2:2 into training, validation and test.

Pre-processing of the data The raw texts have been passed through the following steps:

- Search for contractions and replace them with the correct expansion[4].
- Filtering of all the non-alphabetical characters.

Lemmatization After pre-processing, we create a lemmatization of the texts using spaCy[5].

[4] `https://pypi.org/project/contractions/`
[5] `https://spacy.io/`

Creation of the matrix The input of the LSTM Layer must be a matrix where each word represents an element defined by a set of features (the columns). We decide to not let the Embedding Layer create this matrix because our dataset for training was not big enough but instead we used a pre-trained vector representation taken from Google Code. This database contains a representation of 3 million words and phrases in a space of 300 features and was trained on part of Google News dataset; the vectorization is done using the tool word2vec [7]. Furthermore, we use the dataset provided by [6], which contains a VAD value for nearly 14,000 English lemmas. The scale ranges from 1 (happy [excited; controlled]) to 9 (unhappy [calm; in control]). The corpus was created thanks to manual annotators and the value used in this project correspond to the mean value of the annotations.

For each word/lemma, we take its representation in the pre-trained vectors from Google Code and concatenate it to the representation in terms of VAD. If a word/lemma is not contained in the database of Google Code, we create a vector of random samples from a normal (Gaussian) distribution and if a word is not contained in VAD database, we give it a neutral VAD value (5,5,5).

Based on the mean length of the training set, we decided to fix the length of the text to 15 words (See Fig. 2). (the texts longer are cut and the ones shorter are padded with a standard vector representation).

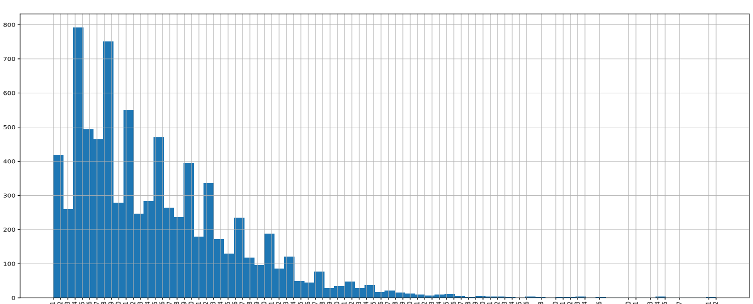

Fig. 2: Histogram of the length of the texts in the training set.

In this way, we obtain a matrix representation of [15, 300] for every text and this is the input of the LSTM Layer.

3.2 Evaluation metrics

As in [3] the metrics used for the evaluation of the model are MAE and r, defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|A_i - P_i| \tag{1}$$

$$r = \frac{1}{n-1} \sum_{i=1}^{n} \left(\frac{A_i - \hat{A}}{\sigma_A} \right) \left(\frac{P_i - \hat{P}}{\sigma_P} \right) \quad (2)$$

where A_i is the actual value, P_i is the predicted value, n is the number of test samples, $\hat{A}$ and $\hat{P}$ denote the arithmetic mean of A and P and σ is the standard deviation. The MAE measures the error rate, and the Pearson correlation coefficient r measures the linear correlation between the actual values and the predicted ones. A lower MAE and a higher r indicate more accurate prediction and therefore better performance.

3.3 Experimental methodology

We used the training set and the validation set to decide the best parameters for the rate of the Dropout Layer, the number of units in the LSTM and the number of units in the first Dense Layer. The best configuration found is: 0.2 rate for the Dropout Layer, 160 units for the LSTM Layer and 25 units for the first Dense Layer. The decision was based on a 10-cross-validation and a Keras Tuner RandomSearch.

3.4 Performance of the model

On the test set the model shows the following metrics:

- Mean Square Error: 0.0717
- MAE: 0.2022
- Pearson Correlation Coefficient r: 0.3398

3.5 Exploit the model to create an emotional profile of a character in a movie

The Cornell Movie-Dialogs Corpus After the model has been trained and tested, we used it to predict the VAD in utterances in film. The Cornell Movie–Dialogs Corpus [6] contains a large metadata-rich collection of fictional conversations extracted from raw movie scripts. It holds 304,713 utterances from 617 movies.

The predictions made by the model Once a film and a character in it are selected, all the utterances of the character are retrieved and pre-processed as for the EmoBank corpus; the texts are pre-processed, is performed lemmatization and the matrix is created. Then, the matrix is passed as input to the pre-trained model and a prediction of VAD values for each utterance is performed.

3.6 Emotional profile of a character

The emotional profile of a character is shown as a boxplot of each dimension (See Fig. 3) in which we can see the mean value, the first and third quartiles, the maximum and the minimum and the outliers.

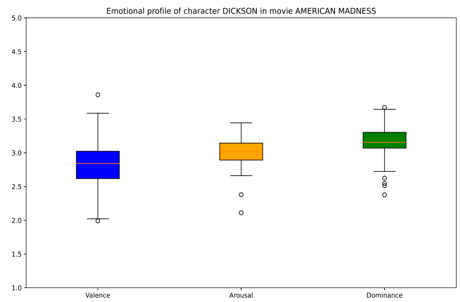

Fig. 3: Sample emotional profile.

The predictions in time We supposed that the lines of a movie can be ordered by the incremental number in the line index. To show how the emotional profile of a character changes in time we plot the VAD of the lines ordered in time. (See Fig. 4)

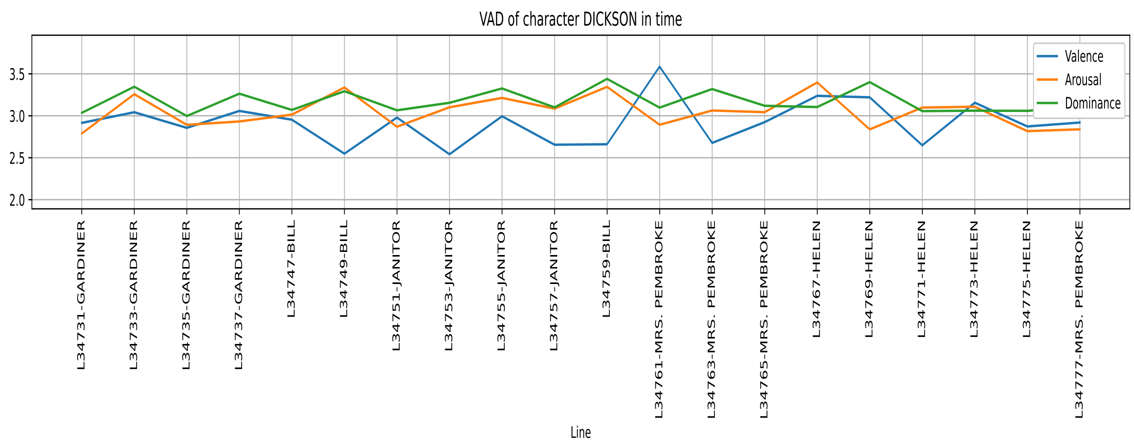

Fig. 4: VAD values of a character in time.

How the emotional profile is affected by the relations among different characters To understand if the emotional status of a character is affected when she/he speaks with another character, we take the mean value of each VAD dimension for the character and we measure how much it differs from the median of the utterances when the main character is speaking only to that character (See Fig. 5).

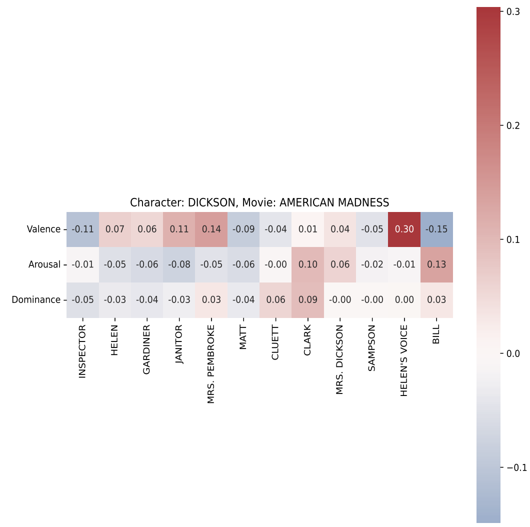

Fig. 5: How the mean VAD values of a character changes in respect to how she/he is talking to.

4 Concluding remarks

We compared the results of this model to the ones obtained in [3] when using the LSTM encoder but, since they evaluated the MAE and r for every dimension (VAD), we took the mean of these values (See Table 1). As we can see in Table 1,

Table 1: Mean values of LSTM in [3] and this model.

Model	MAE	r
Independent	0.25166	0.28066
External	0.193	0.288
Internal	0.18966	0.37266
Combined	0.18766	0.37733
This model	0.2022	0.3398

the results obtained for MAE are better than the independent LSTM model created in [3] and for r better than the independent and external model but worse than the other types of models.

It seems that the model developed in this study is better than the independent model but worst than the others.

There are a great number of other possible configurations of the LSTM model than can be tested to enhance the performance of the models, for instance:

– Integrate the part of speech in the word vectorization.

– Play with the layer of the networks (adding/removing layers).
– Combine the results of different configurations.

Furthermore, it can be interesting to try also the other models used in [3] as CNN, HAN.

References

1. Ekman, P.: An argument for basic emotions. Cogn. Emot. **6** (3-4) 169–200 (1992)
2. Buehel, S. and Hahn, U.: EMOBANK: Studying the Impact of Annotation Perspective and Representation Format on Dimensional Emotion Analysis. Proceedings of the 15th conference of the european chapter of the association for computational linguistics **2**, 578–585 (2017)
3. Xie, H. and Lin, W. and Lin, S. and Wang, J. and Yu, L.-C.: A multi-dimensional relation model for dimensional sentiment analysis. Information Sciences **579**, 832–844 (2021)
4. Warriner, A.B. and Kuperman, V. and Brysbaert, M.: Norms of valence, arousal, and dominance for 13,915 English lemmas. Behavior research methods **45**, pp. 1191–1207 (2013)
5. Aggarwal C.C.: Machine Learning for Text. Springer, Berlin/Heidelberg, Germany (2018)
6. Danescu-Niculescu-Mizil, C. and Lee, L.: Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In: Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics. ACL (2011)
7. Google Code, word2vec, `https://code.google.com/archive/p/word2vec/`. Last accessed 16 Feb 2023