Using Machine and Deep Learning to Promote Mood Prediction(Group 23)*

Haohui Zhang(2722930)[1], Willem Van Der Spek(2607407)[1], and Yilin Li(2737659)[1]

1 Vrije Universiteit Amsterdam, Netherlands
{h17.zhang,f.w.e.van.der.spek,y45.li}@student.vu.nl

Abstract. In this work, machine learning was evaluated for predicting mood based on a data gathered from human behaviour on a smartphone. An extensive approach to data preprocessing, analysis, feature engineering and model selection has been described. Several machine learning methods were evaluated and bench-marked on their performance, demonstrating that the models outperform a naive benchmark.

Keywords: Machine Learning · Neural Network · Behaviour Prediction · Temporal Model

1 Introduction

Sentiment prediction and analysis remains a significant challenge to many business analysts and commercial project when a set of unpredictable variables are highly related to the label. Integration of time-sensitive neural networks and machine learning algorithms is illustrating great promise in extracting valuable feature from raw data to assist the prediction of mood. To implement preprocessing of raw data and deleting the noise before prediction, a number of methods have been proposed, ranging from Pearson Correlation Coefficient [1] to dendogram analysis [2]. After the processed dataset are generated through multiple test, we intend to implement several neural networks and machine learning models to generate accurate prediction of the subject's mood. To demonstrate the time-sensitivity of the data from human behaviour, we propose to report out attempts in utilizing state-of-the-art deep learning models, such as a *Long Short Term Memory* (LSTM) [3] and *Bi-directional LSTM* Bi-LSTM [4]. Besides, attention-based mechanisms are also imported in the process of extracting features, consider more weight parameters will be added to extract more relevant features [5]. What's more, much of machine learning algorithms fits well with low-dimensional features. For instance, Linear Regression and LASSO are treated as widely-used linear regressors which generate prediction based on feature analysis. Compared with a naive benchmark, we also execute multiple non-linear methods (e.g., Random Forest, KNN [6] and SVR [7]) to realize the relationship between human behaviour and sentiment. At last, the *Mean Squared Error* (MSE), *Mean Absolute Error* (MAE) and *R-squared* (R^2) will be utilized for evaluation.

* Supported by Data Mining Techniques, Vrije Universiteit Amsterdam

2 Data Exploration & Preprocessing

2.1 First Glance

The initially provided dataset contains a total of 376912 rows and 4 columns: `id`, `time`, `variable` and `value`. The column `variable` contains all the features that can potentially be used for prediction and hence the total amount of rows in the initial dataset is not representative of the amount of samples that can actually be used. For our initial exploration, we aim to capture how many occurrences there are per variable and how the `id` column is distributed according to its occurrences. We have captured these distributions in figure 1. From a visual observation, it can be concluded that the provided dataset is sparse, e.g. it contains a lot of missing values due to the skew in variable occurrence. This is problematic due to the variables containing missing values being extremely noisy. In particular, we hypothesize the data to follow a *power-law distribution*, a very common phenomenon in real-world data[8]. The power-law equation is formulated as follows:

$$f(x) = kx^{-\alpha} \tag{1}$$

We let x be one of our features represented as a (real) number, k and α be some constants and $f(x)$ the probability of a value being missing or not. The intuition behind this relationship is that a minority of the features holds the majority of valid data points, thus making a lot features possibly redundant.

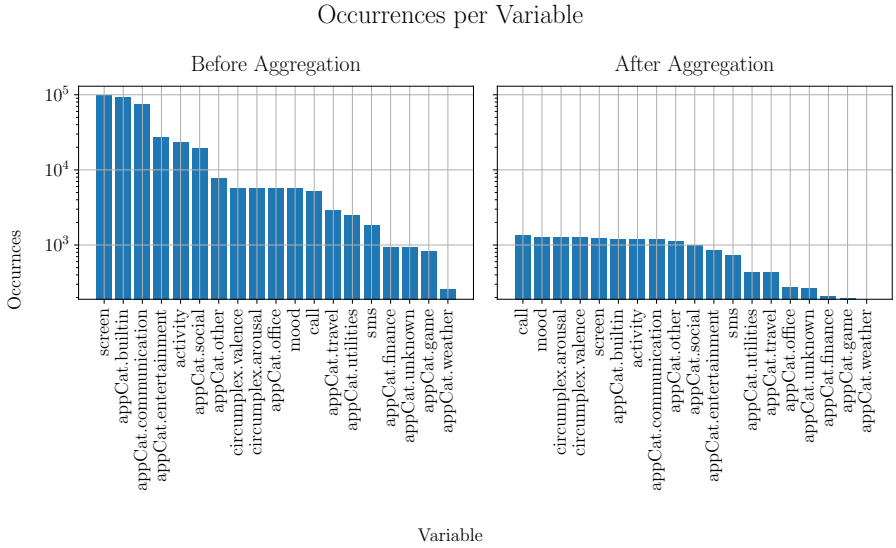

Fig. 1: Distributions for the `variable` and `id` values' occurrences. Note the log scale.

2.2 Preprocessing

The main goal of the data prepossessing pipeline will be to deal with the missing values. As a reference, we provide the entire data prepossessing pipeline in table **??**. Starting off with the initial dataset, first the `variable` and `value` column were unrolled in order to obtain these variables as features in separate columns. Afterwards the `time` column was processed and the dataset was sorted according to this time. Afterwards the data was grouped by the `id` column and all of the following operations will be performed on the grouped data. After the final preprocessing step, the data will be merged back to a single group. Due to the task being to predict the average mood and not the mood per timestamp, the arithmetic mean for the timestamp per day, not only for the *mood* variable, but also for every other variable was computed. Consequently, missing values are now only present when all the data for that particular day is missing. As such, we obtain a new distribution for our variables, which are displayed in figure 1. From a pure visual observation, the occurrence distribution is now far less skewed at the cost of reducing the amount of rows. Then, every row where `mood` is still missing is removed from the dataset, as no predictions can be made on a missing target feature. Also, all features containing more than 40% missing values were discarded due to the noise introduced by these features, this yields the *filtered dataset*.

From subsection 2.3 it becomes apparent that the distributions of some of our features are highly skewed; this means that these features are prone to having outliers. Therefore, to impute the missing the data, we have chosen to use the median of the existing values of the features instead of the mean due to the former one being more statistically robust. Finally, we create our target variable by shifting all of the values of the mood feature one row up and deleting the last row for the dataset per ID. Additionally, an approach is defined in subsection **??** to obtain additional features to improve the predictive power of our dataset. We consider this feature engineered model as final and ready for our models to use. We split the data into 80% for the training set and 20% for the test set and use cross-validation to evaluate the models during development.

2.3 Deeper exploration

After the initial preprocessing step, a deeper dive in the data was taken to draw further insights on the dataset. Considering the sparsity of this dataset, a dendogram displaying the *nullity correlation* between features was created in figure 2. Nullity correlation is a number that represents how strongly the null values of a particular column correlate with those of another column, i.e. how many null values are present for the exact same data-point. We considered nullity correlation to be important due to the intent of having to impute the missing values with actual values. As a result, features with low nullity correlation will have relatively many noisy, imputed values and are assumed to be poorer predictors. The nullity dendogram immediately shows strong nullity correlation between the `mood`, `circumplex.valence` and `circumplex.arousal` features. The cluster

Table 1: Total preprocessing pipeline summarized with the effect of a particular step on the dataset's shape. Note that we summed the rows for every grouped ID.

Name	**Preprocessing step**	**Shape (rows × cols)**
Initial	N.A.	376912×4
Unrolled	Unrolled the `variable` column to several columns as features	376912×20
Grouped	grouped by `id` column	376912×4
Sorted	Converted `time` and sorted by it.	$96{,}578 \times 21$
Aggregated	Aggregated rows to timestamps of the same day	3051×19
Filtered	Removed rows where `mood` is missing as well as columns with too many missing values	1263×12
Imputed	Replaced missing values with imputed values.	1263×12
Final	Added the engineered features.	1263×18

on the left-hand-side of the dendogram is, on the other hand, relatively poorly correlated with the rest of the dataset.

From here, we looked at the *pearson correlation* between the features in the aggregated dataset. The Pearson correlation is a representation of *linear correlation* between to variables. The results for these correlation can be found in figure 4. From this figure it becomes apparent that there is very little correlation to no correlation between almost all of the features. The most notable correlation is between `mood` and `circumplex.valence` showing a moderately high correlation value. Since our target variable has the exact same data points as `mood`, we consider this feature as useful.

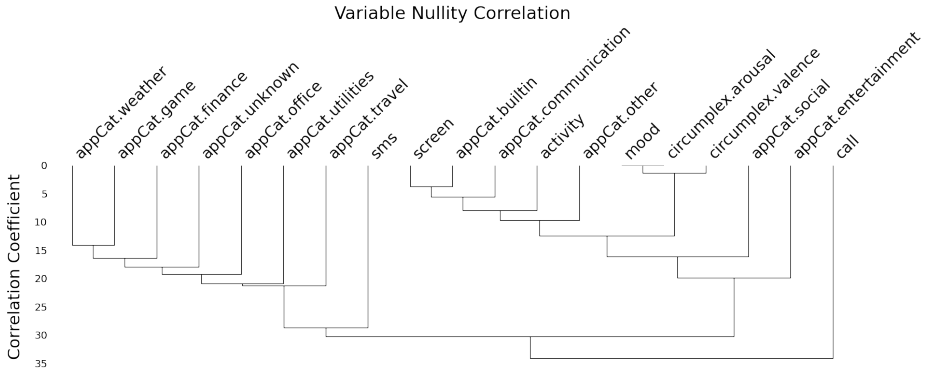

Fig. 2: Dendogram displaying nullity correlation between features. A lower score on the vertical axis implies higher nullity correlation.

Finally, analysis on the distribution of the features of the filtered dataset have been made as displayed in figure 3. Again, we observe highly skewed distributions for most of our features, possibly in accordance with equation 1. On the other hand, the features `mood`, `circumplex.valence` and `circumplex.arousal` are not nearly as skewed as the other features and take a far more similar shape.

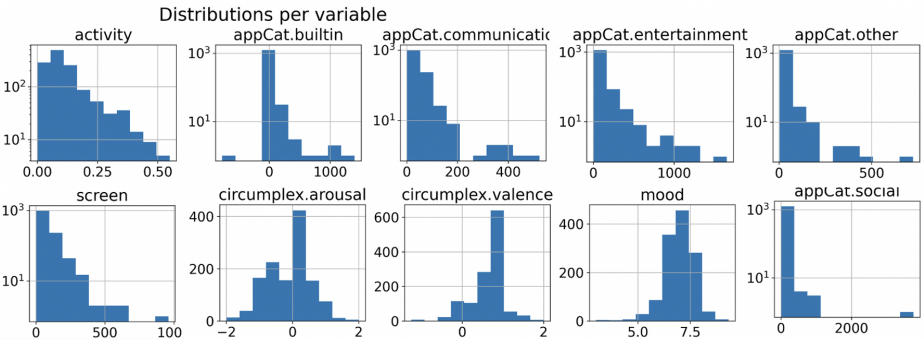

Fig. 3: Histogram for every variable in the dataset. Logarithmic scale on the y axes has been used for some of the variables due to their power-law nature.

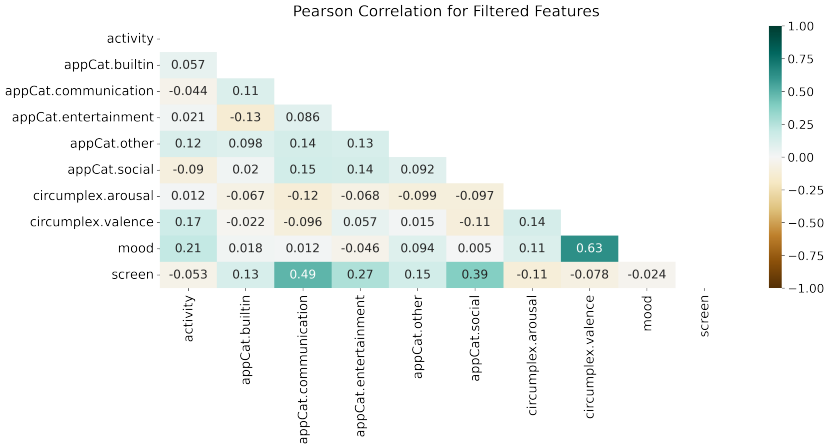

Fig. 4: Pearson correlation between all features in the filtered dataset.

2.4 Feature Engineering

In order to obtain a target variable, we use the `mood` feature and shift all its rows one back so that Based on our data exploration and insights from

these several visualisations, we hypothesize that `mood`, `circumplex.valence` and `circumplex.arousal` will be most crucial to predict our target feature. Hence we have chosen to pick data from the past 5 days of these features and compute the arithmetic mean and standard deviation of this sample. We have obtained the amount of days to look back by trying out different values and evaluating them through cross-validation and found the optimum to be at 5. Again, we perform this operation for every unique ID in the imputed dataset. Note that for the first five rows, there is not sufficient past data and thus we initially represent these values as missing. After computing the entire row, the median of the present mood value is used to fill the missing values for our new features. Thus we obtain a total of seven new features, one of which: `next_mood` being the target feature.

3 Multiple Predictive Models Aided Training and Testing

With the aiming of predicting the value of mood based on five previous days, we intended to establish various predictive models on valid and designed dataset which were generated through implementing all previous steps. An overview for the establishment, training and prediction phases of our approaches is summarized in Fig. 5. The establishment and training phase of various models consists of four steps: 1) extracting feature from input matrix based on machine learning models; 2) analyzing dataset's timing features by designing temporal model; 3) giving a benchmark as the baseline of evaluation; 4) next, six artificially generated features, which are also attached to the processed dataset, will be utilized to train machine learning models and tried to realize whether more features will improve the performance of prediction. Details are shown as follows.

3.1 Establishing Machine Learning Models for Prediction

Considering the strong randomness of the subjects' mood changes, we proposed to select and train a variety of non-linear regressors to extract valuable features from the input matrix and then quantify the subjects' mood for the next day. Also, we utilized several linear regressors as the baseline for evaluation (i.e., to prove whether non-linear regressors are more suitable for this case or not).

In detail, Linear Regression (LR) is first selected as the baseline of training. This is because: 1) LR assumes linear relationship between input and output 2) LR's assumption assist in determining pre-processed dataset is linear-related or non-linear correlation. Then, LASSO Regression, which holds the sum of coefficient values as penalty to avoid prediction errors, is also suitable for this case if we consider the correlation between `circumplex.valence` and `mood`.

Besides, four different non-linear regressors are also selected to fit the train set and generate result to predict the mood in the sixth day. They are: 1) Supported Vector Regression (SVR), which are widely implement in various usage scenario; 2) Random Forest, which utilizes multiple decision trees for generating more fitted prediction but requires more inputs in training; 3) KNN, whose assumption

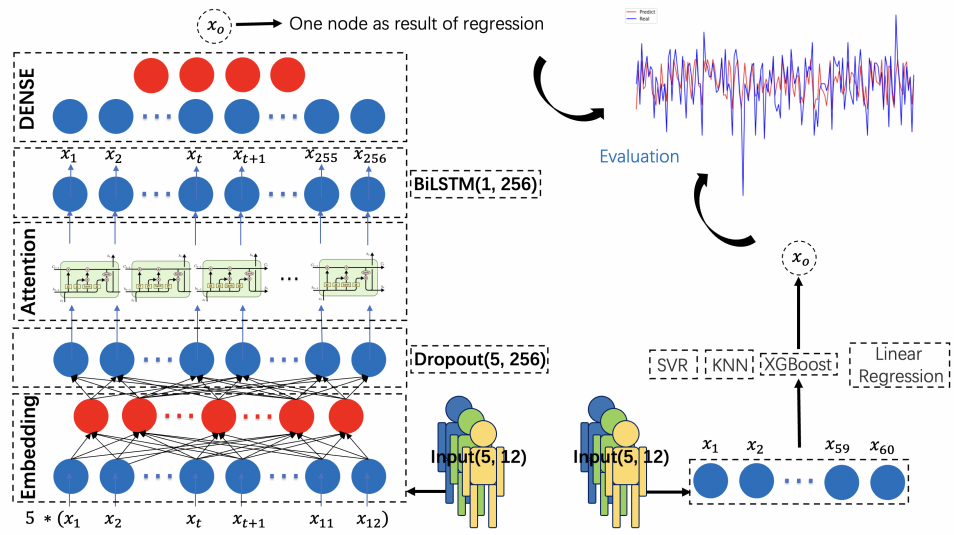

Fig. 5: The overview of the whole pipeline of training and prediction. The left panel shows the process of implementing temporal model while the right one illustrates the steps of training multiple machine learning models.

of prediction is similar to our case; 4)XGBoost, which only requires few features in term of training but is likely to encounter overfitting. Table 2 (See above panel) summarized the fundamental information of multiple machine learning (feature engineered) models.

3.2 Implementing Temporal Models for Prediction

Training a time-sensitive neural network (NN) model with time series as input is also a feasible method for prediction, which is worth to investigate, especially considering that all available data from subjects are collected by date. Multiple NNs are selected and play different roles in the process of extracting time series of features. In general, the establishment of temporal model is divided into 3 steps: 1) importing an embedding layer to map low-dimensional input into a high-dimensional space to expend the features for training; 2) stacking multiple time-sensitive layers (e.g., LSTM, RNN, Bi-LSTM, etc.) to extract timing features with the help of "Dropout" layer; 3) "Dense" layers are utilized to predict the mood of the sixth day. Figure 5 (See left panel) illustrates the pipeline of prediction based on multi-layers temporal model.

In details, during the process of analyzing temporal models, a variety of widely-used neural network models is first selected and evaluated in different combinations. The comparison between LSTM and Bi-LSTM is summarized in the test for various structures of temporal models. Considering that Bi-LSTM leverages future context to learn better representations of single words, we ex-

pected there are only few differences between these two models. However, the import of attention-based mechanism is able to promote the performance of temporal model based on adding more weight parameters and extracting more relevant features. Then, multiple artificially generated features (i.e., mean of five days' mood, etc.) combined with original dataset are trained in the most fitted temporal model to search the potential methods for promoting the accuracy of prediction. Table 2 (See middle panel) summarized the fundamental information of multiple temporal models.

3.3 Designing a Benchmark for Prediction

According to the requirements in guidance, the value of mood in the fifth day will be treated as the prediction of the sixth day. Table 2 (See bottom panel) summarized the fundamental information of the benchmark.

Table 2: An overview of feature engineered, temporal models and benchmark we utilized.

Feature Engineered Models	Parameters	(sample, days∗features)
Linear Regression	N/A	Input:(1263, 5∗12)
LASSO	N/A	Input:(1263, 5∗12)
Random Forest	N/A	Input:(1263, 5∗12)
KNN	n_neighbor=5	Input:(1263, 5∗12)
Supported Vector Regression	kernel = 'rbf'	Input:(1263, 5∗12)
XGBoost	n_esti..=50,max_depth=12	Input:(1263, 5∗12)
Temporal Models	Shape in multi-layers	(sample,days,features)
LSTM+Dropout	5∗128,5∗128,5∗1,1∗1	Input:(1263, 5, 12)
Bi-LSTM∗2+Dropout	5∗12,5∗256,1∗256,1∗4,1∗1	Input:(1263, 5, 12)
Bi-LSTM+Attention+Dropout	5∗12,5∗256,1∗64,1∗4,1∗1	Input:(1263, 5, 12)
Bi-LSTM∗2+Attention+Dropout	5∗12,5∗256,1∗256,1∗1	Input:(1263, 5, 12)
Benchmark	Description	(sample,features)
The Fifth Day	The mood of the fifth day	Input:(1263, 12)
Improved Temporal Model	Description	(sample, features)
Temporal Models	Add 6 More Features	Input:(1263, 5, 12)

4 Result and Discussion

4.1 Analyzing Results Using Statistics

In order to analyze our results, we aim to use the (1) MSE, the arithmetic mean of the second power of prediction errors, (2) MAE, being the arithmetic mean of the prediction errors and (3) the R^2 value, Intuitively, the R^2 shows how well a model performs compared to a baseline: using the mean of all training features for every prediction. The mean absolute error provides an intuitive result; the

average error to expect from the model's prediction. We use the mean squared error due to its stricter penalty on greater errors. Finally, the R^2 value gives us an estimate how well the model accounts for variance in this particular dataset.

Table 3 shows the performance of all tested models. Due to the dataset we select is a small-scale one (i.e., only contains 1263 records) and the lack of high-dimensional features, all temporal models based on neural networks obtained a relatively poor performance while most of machine learning models got more accurate prediction of mood. It is also worth to note that high-dimensional features are more suitable for neural networks if we consider the improved result (See Table 3, the bottom panel) with six more generated features.

Table 3: The performance of feature engineered, temporal models and benchmark we tested. Note: all the following tests implement 8-folds cross validation.

Feature Engineered Models	MSE	MAE	R^2
Linear Regression	0.4090	0.4724	0.2581
LASSO	**0.3994**	**0.4650**	**0.2582**
SVR	0.7483	0.6386	0.2402
KNN	0.7261	0.6383	-0.3528
RF	0.4154	0.4740	0.2235
XGBoost	0.4592	0.4963	0.1430
Temporal Models			
LSTM+Dropout	0.9986	N/A	N/A
Bi-LSTM∗2+Dropout	0.7836	0.6505	0.0738
Bi-LSTM+Attention+Dropout	0.8016	0.6544	0.0762
Bi-LSTM∗2+Attention+Dropout	0.7554	0.6407	0.1934
Benchmark			
The Fifth Day	0.5706	0.5513	-0.0494
More Features Aided Prediction			
Bi-LSTM∗2+Attention+Dropout	0.7357	0.6310	0.1458

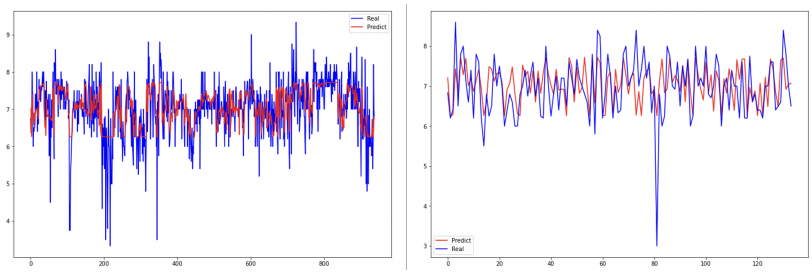

Fig. 6: The overview of the comparison between predictions and labels. The left panel shows the trend-line of train set while the right one illustrates the trend-line of test set.

4.2 Discussion

From our results we can infer that LASSO has the best performance out of all our models, with linear regression and the random forest producing similar results. All of these models outperform the baseline model in terms of all evaluation metrics. We hypothesize these models to have the best performance partially due to the moderate correlation found between `circumplex.valence` and `mood`, which makes linear models the best predictors in a relative sense. Furthermore, we suspect the temporal models to perform relatively worse due to not enough data being present to account for the large amount of parameters that they harbor; more complex models require more data to learn patterns in the dataset [9]. Another things worth mentioning is the relatively poor R^2 scores. Since the data was obtained from human behaviour, it is inherently noisy and therefore difficult to predict. Many studies in social sciences report relatively lower R^2 scores than the exact fields due to this noise which explains the relatively low R^2 scores [10]. Nonetheless, some of our models do outperform the baseline model that uses the mood of the past 5 days to predict the mood of the next day due to the feature engineered dataset in conjunction with a relatively simple model having better predictive power.

In total, the pros and cons of different approaches are shown as follows: 1) Temporal Model, such as LSTM and Bi-LSTM, can rapidly fits time-sensitive features. But it is only suitable for training high-dimensional features due to the bottleneck of overfitting; 2) Machine Learning Model, such as LR and SVR, is sensitive to the quality of features but can extract features without limitation of dimension.

5 Conclusion

The original goal of this work was to evaluate the performance of machine learning to predict a person's mood based on a submitted numerical value. Our work describes a detailed approach to preprocess the dataset in order to achieve this. Additionally, we provide a feature engineering approach to improve this dataset. We continue to describe several used models, along with configurations used for them and benchmark them against a naive regressor that simply always predicts the mood to be that of the fifth day in the dataset, where were able to achieve an MSE of 0.40 and an R^2 score of 0.26, whereas the benchmark yielded an MSE of 0.57. These results suggest that our best models are better than a naive classifier. However, the more complex neural networks were not always able to outperform the naive benchmarks due to these models being too complex and large for the given dataset.

When enhanced with more data, we believe our approach will eventually yield even better results, especially for the deep learning methods. Additionally, extra methods for feature engineering and their effect on performance could be evaluated.

Bibliography

[1] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*. Springer, 2009, pp. 1–4.

[2] T. CaliŃski, "Dendrogram," *Wiley StatsRef: Statistics Reference Online*, 2014.

[3] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying lstm to time series predictable through time-window approaches," in *Neural Nets WIRN Vietri-01*. Springer, 2002, pp. 193–200.

[4] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[5] S. Wang and J. Jiang, "Learning natural language inference with lstm," *arXiv preprint arXiv:1512.08849*, 2015.

[6] S. B. Imandoust, M. Bolandraftar *et al.*, "Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background," *International Journal of Engineering Research and Applications*, vol. 3, no. 5, pp. 605–610, 2013.

[7] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-svr for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert systems with applications*, vol. 36, no. 3, pp. 6164–6173, 2009.

[8] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Review*, vol. 51, no. 4, pp. 661–703, 2009. [Online]. Available: http://www.jstor.org/stable/25662336

[9] X. Zhu, C. Vondrick, C. C. Fowlkes, and D. Ramanan, "Do we need more training data?" *International Journal of Computer Vision*, vol. 119, no. 1, pp. 76–92, 2016.

[10] F. Moksony, "Small is beautiful: The use and interpretation of r2 in social research," *Szociologiai Szemle*, pp. 130–138, 01 1999.