Twitter, Stock prices and Sentiments: Predicting the stock price of Twitter Inc using a Sentiment Analysis method

Bertrand Kwibuka
Economics Department
Ashoka University

Suchir Kalra
Economics Department
Ashoka University

***Abstract*—**

We predict the stock price of Twitter Inc. using sentiment analysis and a recurrent neural network (RNN) architecture. We specifically use the long short-term memory neural network (LSTM) with varying hyperparameters to experiment with the used dataset. The models are trained on data collected from various news sources and a repository of stock indexes. The news corpus' sentiments are collected in two setups and the stock prices are predicted. Our study shows that sentiment analysis done with the news text separated by dots performs better at an R^2 Score of 93.2% than other sentiment analysis done without separating the news text at an R^2 score of 92.5% . Overall, both the Sentiment analysis setups have a much better accuracy compared to the setup without sentiment analysis, which has an R^2 score of 77.3%. Our project demonstrates that LSTM is a powerful neural network when predicting stock prices using sentiments for a social media company.

***Index Terms*—LSTM, News Text Description, Stock Prices, TWTR, Split, Unsplit**

I. Introduction and Motivation

Stock markets have become a crucial part of any corporation. The state of a corporation's stock price could reflect its performance, corporate culture, work structure, people's perceptions, and more. However, the stock market is extremely volatile, which is the extent to which a trading price series fluctuates over time. Investors and speculators find it very hard to accurately predict the stock prices due to the lack of information and also the inability to humanely process the data available.

Moreover, external factors such as political situations, global economic conditions, financial performance, and pandemics make it difficult to predict the stock market due to the vast uncertainty in the economic structure. Apart from these external factors, other physical and psychological factors such as rational or irrational behavior of the investors also determine the prediction process, its outcome, and sometimes lead to less accurate stock market predictions. By using automated algorithmic methods to evaluate the stock market, machine learning can be used to detect stock market trends.

In this project we focused on Twitter Inc. This is a company that solely produces intangible outputs, meaning digital-related content and products. Without associating this with external subsidiaries of the company, we suspect that the majority of its stock market price performance relies on its primary product which is a micro-blogging and social networking service on which users post and interact with messages known as "tweets". There is solid evidence that twitter stock price is severely affected by news about the company Twitter Inc and this provides general information about how social media companies are very careful about how they are represented in the media. [3] [8]

Therefore, with increased usage of Twitter globally, opinions about Twitter in news articles are suspected to cause an increased effect on the stock prices of Twitter. For instance, very recently, the long discussions and evaluations around the corporation since Elon Musk's takeover of Twitter. Elon Musk took Twitter private and off the stock market at a deal of $\$44$ billion on 26th April 2022 [9].

In this paper, we have predicted the stock prices of Twitter Inc (TWTR) using a sentiment method with the LSTM (Long Short Term Memory) model for the period of 2013-2022. We chose 2013 as the starting period in the time-range because that is when Twitter had its initial public offering.

The sentiment analysis method considers the sentiments in a vast range of news articles and headlines about Twitter to predict the stock prices of Twitter. We also use a non-sentiment analysis that does not consider the news headlines and predicts the stock prices using the past prices. When sentiments toward Twitter are positive, the stock prices would tend to go up and when sentiments towards Twitter are negative, the stock prices would go down.

II. Related Work

The use of sentiment analysis in stock market prediction is a very active research field and is an increasingly important field due to the 'nonlinear' nature of the stock market. Most of the existing literature revolves around the use of Tweets and Twitter Sentiment analysis in predicting the stock prices of corporations like Apple and others. [4]

Recurrent back propagation takes a very long time to learn to store information over extended time intervals, mostly because decaying error back flow is insufficient. By analyzing Hochreiter's 1991 [1] analysis of this problem, Hochreiter (1997)[2] solved it by initiating the novel, efficient, gradient-based method called "Long Short-Term Memory" (LSTM). The experimentation with real datasets showed that LSTM was computationally more efficient than the then state-of-the-art methods such as RTRL, BPTT, Elman nets and others.

Kalyani et al. (2016) [4] studies the relationship between news and stock trend of Apple Inc. They precisely use three classification models - RF(Random Forest), SVM (Support Vector Machine), and Naive Bayes. The authors show that RF(Random Forest) and SVM(support vector machine) give good and encouraging results compared to the Naive Bayes Model. They show that their prediction model has increased the accuracy by 30% relative to the news random labeling.

In Nagar et al.(2012) [7], the time variation of News Sentiment towards AAPL (Apple Corporation) shows a very strong correlation with the actual stock price movement of Apple (Using NLP techniques). A drawback in their research is that they are merely basing their method on the polarity of positive and negative polarity words as a measure of the sentiment rather than taking into account the entire description or overall sentiment of the text.

Na et al.(2021) [6] have employed a deep neural network (*ANN - Artificial Neural Network*) in order to predict the stock prices on the basis of informed traders' activities. They show that the probability of their model being correct could be as high as 74%.

III. Experimental Setup

In this project, we use two different datasets that are merged to make a final dataset used for training and testing. We constructed the first dataset using publicly available NYSE index (New York Stock Exchange) data on a company called Twitter Inc. **TWTR** is the ticker symbol on the NYSE index. Twitter Inc., the company behind the platform, is headquartered in San Francisco, California, and has over 25 offices worldwide.

To download this data, we used a specific API of Yahoo Finance, an online repository which collects data on various market indices. The financial data collected was historical time series data ranging from the time Twitter had its initial public offering (IPO) in November 2013 till February 2022. The logic behind the lengthy time range was to account for all the time Twitter had the most sensitive news about its company or its policies in general. This includes the controversy that surrounded the 2016 U.S elections, the 2020 U.S elections, and the Covid-19 pandemic period. All these major events led to Twitter taking extreme measures regarding how it moderates content on its platform and this in turn led to major news outlets having various opinions about Twitter. The data had attributes about time such as date and attributes about stock prices like daily open, daily high, daily low, daily close, and daily volume. For the sake of notation, let's call this dataset A. Figure 1 shows how the daily adjusted Twitter stock prices changed from 2013 up to 2022.

The second dataset consists of a corpus of text from major news outlets. This data was collected using Google News as the main search base for the news. The search keywords were made specific so that results could return news only about Twitter as a platform in the time range of 2013 to 2022. After getting results from Google News, we used a simple tool called *SimpleScrapper* that allowed us to scrape news from 34 news sources including Bloomberg, CNBC, Financial post, etc. The idea behind collecting news from different sources was to consider the fact that a lot of news companies follow an agenda or a general policy about being either leaning to the left, right, centre, etc. Therefore, it was important to collect news from various sources to ensure that our news' data point of view is balanced. We collected headlines or titles, and a major description about the specific news' title. It is important to reiterate that all these headlines or titles were news about Twitter. For the sake of notation, let's call this dataset B.

A. Data preprocessing

We did our pre-processing by using the datasets in pickle format for easier and faster operations. Pre-processing was divided into 3 parts. The pre-processing done on dataset A, dataset B, and dataset A and B combined. Let's call the combined dataset C.

Dataset A has attributes about time such as date and attributes about stock prices such as daily open, daily high, daily low, daily close, daily volume. We wanted to keep time and adjusted daily closing prices. Therefore, we removed all unnecessary columns and kept two essential columns for the study which are time and adjusted daily closing prices of the Twitter stock. After this process, dataset A was saved again in pickle format to be used later.

Dataset B has attributes about time, news sources, news titles, and news description. The first process in the pre-processing part is to split sentences of news description

by splitting them following where they stop by dot, then specifying that the data is a time series data for indexing, and then removing news rows which had time in a format of 'x days/hours ago' to keep only rows with a format that can be used in python (dd-mm-yyyy). The next pre-processing of this dataset was to assign mean polarity scores on the split text so that these scores can be used later in the model. The second part of the pre-processing process for dataset B was to assign polarity scores on the entire segments of news description without splitting them by dot to compare results later in the model. It is important to notice that what we refer to as news text description is a paragraph made of 2 to 4 sentences describing in general what the article is about. It has a sense of being a summary of the elongated news text.

After processing dataset A and B, we needed to merge them to have final features that will be fed to the model. The merging process consisted of taking elements from dataset A and adding them to dataset B as a column. Here, it is important to notice that since we are doing a study on stock price prediction using sentiments, in a case where we want to predict the stock price for day Y, we had to use the sentiments from day Y-1 or the previous day. Therefore, the dataset was merged accordingly so that the features fed in the model respect this notion.

The important features of the final dataset C consisted of time as index (time), polarity scores assigned to text split by dot, polarity scores assigned to the entire news description, and stock prices. These were the major features used in the model.

B. The model

In this project, we use a Long Short Term Memory (LSTM) neural network to build a model to predict the stock prices of Twitter. LSTM is an artificial recurrent neural network (RNN) architecture used in the field of deep learning which uses feedback connections instead of feedforward neural networks. LSTM is commonly used for processing and predicting time series data. In addition to single data points, LSTM can also process entire sequences of data (for example, audio and video). An example would be the recognition of unsegmented, connected handwriting, and speech recognition. Time series analysis is made easier with LSTM because it is capable of capturing historical trends and predicting future values more accurately. Referring to figure 2, on a brief note, this is how the LSTM model functions:

1) In LSTM, the first step is to decide which information should be omitted from the cell in that particular step. This decision is made using a Sigmoid function. It computes the function based on the previous state $(ht1)$ and the current input xt
2) In the second layer, there are two functions. In the first case, it is the 'sigmoid function', and the second, it is the $tanh$ function. By using the sigmoid function, we can decide which values to let through 0 or 1. By using the $tanh$ function, the values passed are weighted according to their importance, ranging from 1 to 1.
3) In the third step, we determine what the final output will be. To begin, you must run a sigmoid layer that determines which parts of the cell state make it to the output. This means that the cell state must be put through the $tanh$ function so that the values are pushed between 1 and 1 and multiplied by the output of the sigmoid gate.

Data from dataset C was split and we used 80% of the data for training, 20% for testing.

In this project, we also wanted to figure out how performance compares when we use different numbers of neurons. In this case, we used the model in 3 different experiments as follows:

1) Predicting stock prices without using sentiments values
2) Predicting stock prices by using sentiment values from unsplit entire news description text
3) Predicting stock prices by using sentiment values from text split by dot

We specified the performance measure to be done using statistical methods RMSE(Root Mean Square Error), MAPE (Mean absolute percentage error), and R-squared. RMSE and MAPE were key for this study because RMSE indicates the difference between predicted and true values, while MAPE indicates the difference relative to the true values.

$$\text{MAPE} = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{At-Ft}{At}\right|$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(At-Ft)^2}$$

where N = the number of time points, At = the actual / true stock price, Ft = the predicted / forecast value

Before feeding the data in the model, there is a small part of pre-processing which was done. We used the 'MinMaxScaler' function to normalize the data of stock prices. The numeric columns in this dataset have different ranges of values, and normalization is used to scale up these values to a common scale without distorting the ranges. This is also because the model doesn't know exactly how this data is distributed.

IV. Results

A. Model without sentiment values

We have this part in our experiment to see how good is LSTM in predicting stock prices without using sentiment text. In this model, it is important to mention that our results resulted from experimenting with the number of LSTM units (or neurons). These units define the dimension of hidden states (or outputs) and the number of parameters in the LSTM hidden layer.

In this case where we predict stock prices without sentiments, we used the LSTM with 200 units and a dropout

of 20%. The results in this case don't show any indication of over-fitting or under-fitting, therefore leaving out arguments related to bias. The RMSE is 0.039, the Mean Absolute Error is 0.030, the R-square is 0.773.

Referring to figures 3, 4, and 5, the results show a fairly competitive LSTM model which can handle time series data and do a pretty good job in predicting future stock prices. We argue that increasing the number of units can lead to a better performance as shown in other sections below.

B. Model with sentiment values of the un-split news description text

In this analysis, the features were both sentiments (used to predict stock prices of the next day) and stock prices. Starting with a 200 units model, the performance was very poor and by strategically incrementing these units, we concluded that the model where used a range between 4000 and 8000 performed with a reasonably better performance. At first we used 4000 and later used a double of that (8000) to check how the performance improves. The dropout used in both cases is kept at 20%.

Referring to figures 6,7,8, results for the model with 4000 units do not indicate over-fitting or under-fitting. In this case, RMSE is 0.036, Mean Absolute Error is 0.030, and R-square is 0.925. All values are rounded off to 3 decimal places.

To compare these results by adjusting parameters, we run the same model with 8000 units. Referring to figures 12, 13, results for the model with 8000 units also do not indicate over-fitting or under-fitting. In this case, RMSE is 0.034 , Mean Absolute Error is 0.032, and R-square is 0.932.

C. Model with sentiment values of text split by dot

In this analysis, the features were both sentiments (used to predict stock prices of the next day) and stock prices. The model used 4000 units at first and later on used a double of that - 8000- to check the performance of the results. The dropout used in both cases is kept at 20%.

Referring to figures 9,10,and 11, results for the model of 4000 units do not indicate over-fitting or under-fitting. In this case, RMSE is 0.037, Mean Absolute Error is 0.029, and R-square is 0.932. All values are rounded off to 3 decimal places.

Again to compare these results by adjusting parameters, we run the same model with 8000 units. Referring to figures 14, 15, results for this model with 8000 units also do not indicate over-fitting or under-fitting. In this case, RMSE is 0.033, Mean Absolute Error is 0.034 , and R-square is 0.941.

D. Interpretation of results

Referring to figure 16, results from these models indicate that LSTM is indeed a powerful tool to predict the stock market. The novelty of this is that this study studies a company which does not have a lot of varying products leading to very diverging sentiments. This is primarily from the fact that Twitter Inc is not a conglomerate, but a company running only a digital platform as its service and an ad-revenue based business model. Its entire financial model which leads to its valuation and stock prices can be severely affected by news about the company and the majority of this news content is related to the digital platform Twitter.

We can see that the stock prediction without using sentiment analysis has a lower performance than all the other predictions done by using sentiment values. Among the models done by using sentiment analysis, the analysis done on news text split by dot performs much better than the analysis done using sentiments assigned to the entire text of news description. As part of the conclusion, we identify two major outputs. LSTM is good at predicting prices using sentiment analysis and we observe that more LSTM units (leading to a greater dimension of hidden states) is useful in helping the network remember more complex patterns.

V. Conclusion and future work

One limitation of the project is the volatility of stock markets. Daily returns are highly volatile. Thus, making it harder for the neural network to be more accurate when the stock prices have been highly volatile.

This project reviews the performance of LSTM in predicting stock price values of a social media company which has intangible services. In this case, we study the trajectory of Twitter Inc's stock prices from the time it offered its initial public offering till early 2022 using stock prices data and news from various sources. Our results indicate that LSTM is a powerful recurrent neural network capable of working on time series data and sentiment analysis values. Moreover, the performance significantly improves when the sentiments towards Twitter Inc. in the news is used.

In future work, various adaptations on LSTM such as combining CNN with LSTM [10], adding various dimensions to the LSTM model [11] adding differential privacy [5] can be explored to see a much better performance.

References

[1] Josef Hochreiter. "Untersuchungen zu dynamischen neuronalen Netzen". In: *DIPLOMARBEIT IM FACH INFORMATIK* (June 1991).

[2] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural Computation* 9.8 (Nov. 1997). DOI: 10.1162/neco.1997.9.8.1735.

[3] Mathew Ingram. *Here's the one thing that's holding up a potential Twitter acquisition*. Sept. 2016. URL: https://fortune.com/2016/09/07/twitter-acquisition-holdup/.

[4] Joshi Kalyani, Prof. H. N. Bharathi, and Prof. Rao Jyothi. *Stock trend prediction using news sentiment analysis*. 2016. URL: https://arxiv.org/abs/1607.01958.

[5] Xinyi Li et al. "DP-LSTM: Differential privacy-inspired LSTM for stock prediction using financial news". In: *arXiv preprint arXiv:1912.10806* (2019).

[6] Haejung Na and Soonho Kim. "Predicting stock prices based on informed traders' activities using Deep Neural Networks". In: *Economics Letters* 204 (May 2021). DOI: 10.1016/j.econlet.2021.109917.

[7] Anurag Nagar and Michael Hahsler. *Using Text and Data Mining Techniques to extract Stock Market Sentiment from Live News Streams*. 2012.

[8] Lucinda Shen. *Here's why Twitter's stock is plunging*. Oct. 2016. URL: https://fortune.com/2016/10/15/twitter-stock-price-salesforce/.

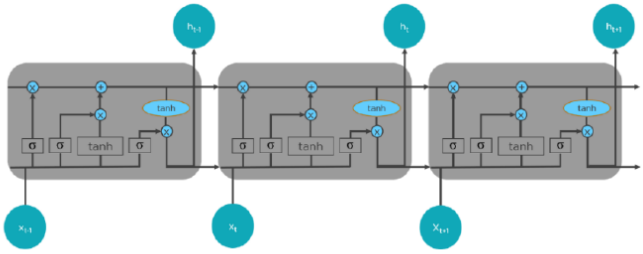
[9] Kurt Wagner. *Elon Musk Lands Deal to Take Twitter Private for 44Billion*. Apr. 2022. URL: https://www.bloomberg.com/news/articles/2022-04-25/elon-musk-clinches-deal-to-take-twitter-private-for-44-billion.

[10] Jin Wang et al. "Dimensional sentiment analysis using a regional CNN-LSTM model". In: *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*. 2016, pp. 225–230.

[11] Jianfeng Zhao, Xia Mao, and Lijiang Chen. "Speech emotion recognition using deep 1D & 2D CNN LSTM networks". In: *Biomedical signal processing and control* 47 (2019), pp. 312–323.

VI. Tables and figures

Fig. 1. Adjusted Stock Prices of Twitter (2013-2022).



Schematic of LSTM method. Source: Google Images

Fig. 2. The architecture of LSTM

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 200)               166400

 dropout (Dropout)           (None, 200)               0

 dense (Dense)               (None, 1)                 201

=================================================================
Total params: 166,601
Trainable params: 166,601
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3. Summary of the model without sentiments

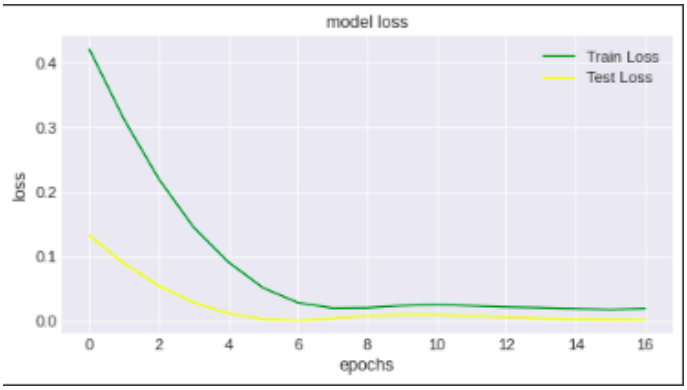


Fig. 4. Learning curve of the model without sentiments

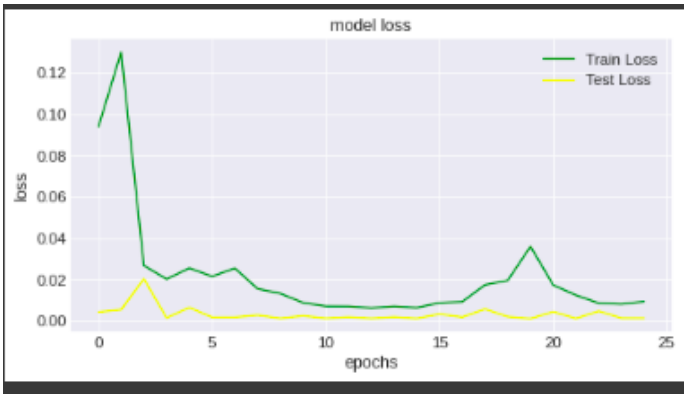


Fig. 7. Learning curve the model with unsplit sentiments split

Fig. 5. LSTM fit for the model without sentiments

Fig. 8. LSTM fit of the model with unsplit sentiments

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_2 (LSTM)               (None, 4000)              64128000

 dropout_2 (Dropout)         (None, 4000)              0

 dense_2 (Dense)             (None, 1)                 4001

=================================================================
Total params: 64,132,001
Trainable params: 64,132,001
Non-trainable params: 0
```

Fig. 6. Summary of the model with unsplit sentiments

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_1 (LSTM)               (None, 4000)              64128000

 dropout_1 (Dropout)         (None, 4000)              0

 dense_1 (Dense)             (None, 1)                 4001

=================================================================
Total params: 64,132,001
Trainable params: 64,132,001
Non-trainable params: 0
_________________________________________________________________
```

Fig. 9. Summary of the model with sentiments split by dot

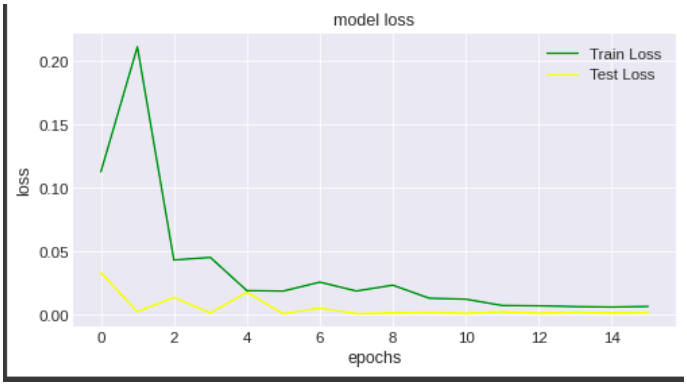


Fig. 10. Learning curve the model with sentiments split by dot

Fig. 11. LSTM fit of the model with sentiments split by dot

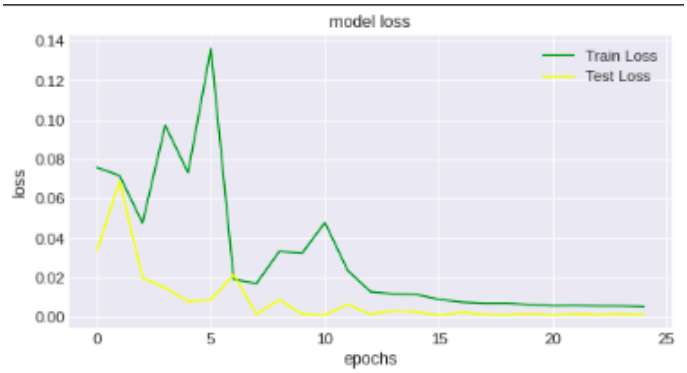


Fig. 12. Learning curve of the model with unsplit sentiments (8000 LSTM units)

Fig. 13. LSTM fit of the model with unsplit sentiments (8000 LSTM units)

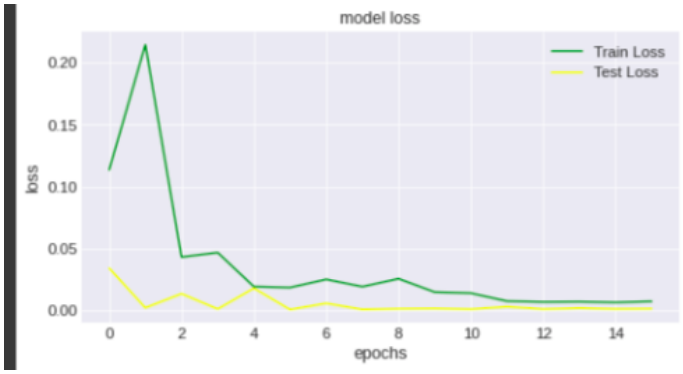


Fig. 14. Learning curve the model with sentiments split by dot (8000 LSTM units)

Fig. 15. LSTM fit of the model with sentiments split by dot (8000 LSTM units)

Model Setup	LSTM Units	Root Mean Square Error	Mean Absolute Error	R-square
Split by dot	4000	0.03739876	0.029095178	0.932
Without splitting text	4000	0.03598226	0.03019154	0.925
Split by dot	8000	0.03310315	0.03450821	0.941
Without splitting text	8000	0.03485774	0.032996196	0.932
Without Sentiments	200	0.038816422	0.030354753	0.773

Fig. 16. Results summary for all models)