

The background features a complex network of thin grey lines and dots, forming a web-like structure. Scattered throughout are various triangles of different sizes and orientations, some with solid black dots at their vertices. The overall aesthetic is modern and technical.

# **Sentiment Analysis on IMDb Reviews**

---

Emmanouil Lykos  
University of Piraeus -NCSR Demokritos



## **INTRODUCTION**

Task Definition and  
Applications

**01**

## **MACHINE LEARNING APPROACH**

Preprocessing + Feature  
Extraction + Training  
algorithms

**02**

## **LONG SHORT-TERM MEMORY NETWORKS**

**03**

# **TABLE OF CONTENTS**

**04**

## **TRANSFER LEARNING**

Incorporate BERT to our  
task.

**05**

## **EXPERIMENTAL EVALUATION**

**06**

## **CONCLUSION & FURTHER WORK**



# 01

# INTRODUCTION

---

Task Definition and Applications



# SENTIMENT ANALYSIS

- Sentiment Analysis is the Natural Language Processing task of identifying the polarity of some text as positive, negative or neutral.
- This task has many real-world applications like:
  - Identifying trends on products, or just political developments.
  - Marketing research.
  - Feedback retrieval.
- This could have been done by retrieving feedback from customers like ratings or forms but it has two major drawbacks:
  - The data that would have collected will be less than the ones retrieved from APIs or crawlers because forums are more accessible than forms.
  - We need people to analyze those data.

***So, this is the point that Machine and Deep Learning techniques enter the game.***





# 02

## MACHINE LEARNING APPROACH

---

Preprocessing + Feature Extraction + Training algorithms

# PREPROCESSING

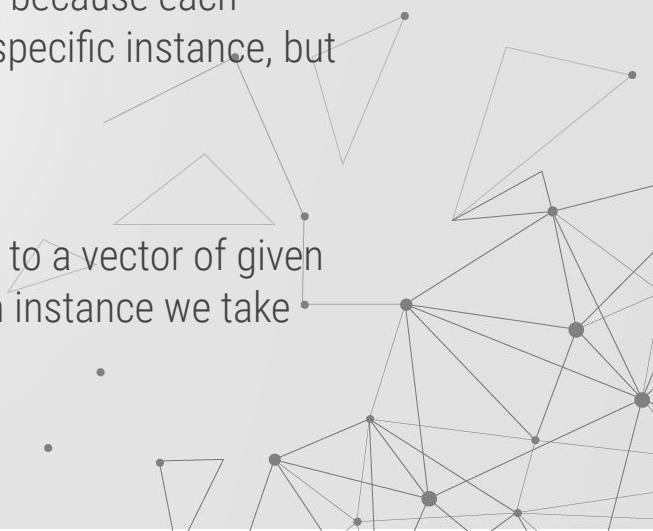
The basic Preprocessing steps that we do in the original data are the following:

1. Lowercase each text instance.
2. Replace more than one sequent whitespaces with one.
3. Tokenize each instance and remove its stopwords. This is done using NLTK library.
4. Convert each token to its root form(stemming).



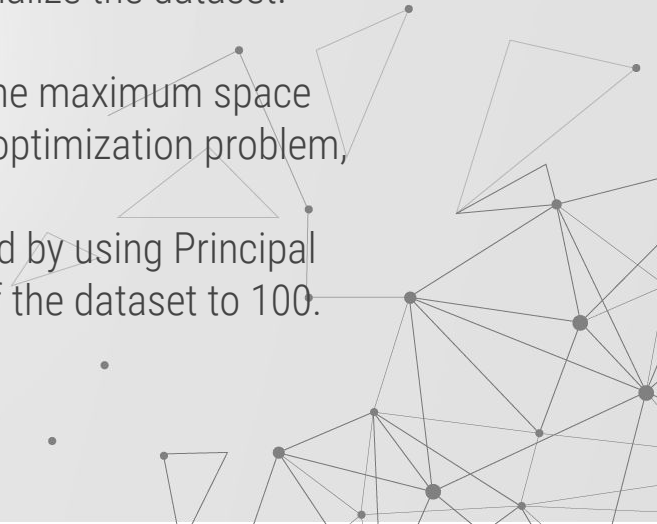
# FEATURE EXTRACTION

- ❖ **Bag of Words:** It is a simple technique that each coordinate of every feature vector represent the number of times that a specific word occurs in the document.
- ❖ **TF-IDF:** This is a more robust technique than Bag of Words because each feature does not consider only the times that appears in a specific instance, but also the times that appears in all documents.
- ❖ **Word2Vec:** This is a pretrained model that converts a word to a vector of given size. In our project, we use vectors of size 300 and for each instance we take the average vector of its words.



# CLASSIFICATION

- ❖ We will use two classifiers. The Naive Bayes and the Support Vector Machines.
- ❖ **Naive Bayes** is an algorithm that wants to calculate the conditional probabilities that each instance belongs to a specific class. This was achieved by using the Bayes theorem along with the assumption that the value of each feature is independent from the others.
  - Before passing the data to Naive Bayes, we just normalize the dataset.
- ❖ **Support Vector Machines** is a method that wants to find the maximum space hyperplane that separates the dataset. This is solved as a optimization problem, thus its slower than Naive Bayes.
  - Before passing the data we normalize the dataset and by using Principal Component Analysis we reduce the dimensionality of the dataset to 100.





# 03

## LSTMs

---

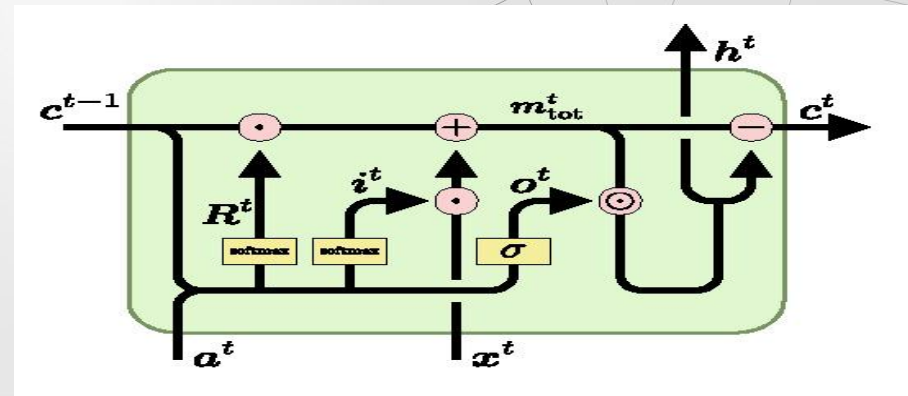
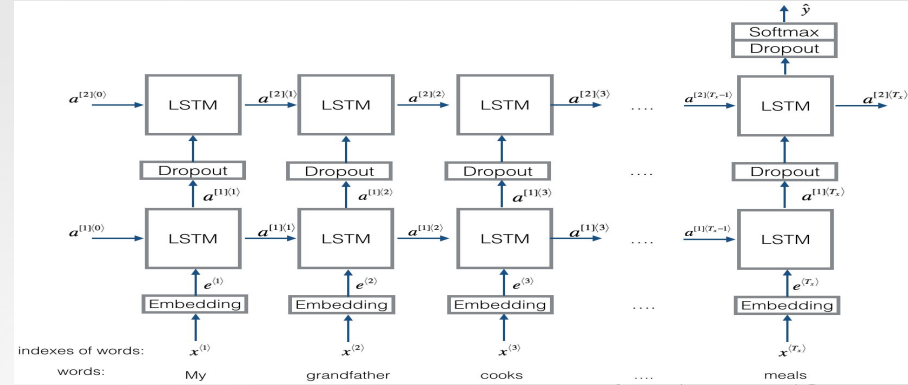
# INTRODUCTION

- ❖ The main drawback of Traditional Machine Learning is that we cannot retrieve the information of the structure of each instance because we use aggregates like histograms and average.
- ❖ Even if change our feature extraction to just concatenate the word embeddings we should fix the number of words that we can take, hence we will still lose information.
- ❖ One approach to fix this problem is to use Sequential Neural Networks.
  - Those networks are designed to handle sequential data like timeseries and text.
  - However, they suffer from the vanishing gradients problem therefore we will use the LSTMs because this is a model that somehow addresses that problem.



# LSTM

- ❖ In the above picture we can see the basic architecture of our model that uses the LSTM layers. However, in our architecture we do not use dropout and we have only 1, one-directional LSTM layer.
- ❖ The below picture shows the architecture of an LSTM cell.
- ❖ In terms of preprocessing we do what we did previously without stopwords removal and stemming and also we add special tokens that denote the start and end of the sentence.



# 04

## TRANSFER LEARNING

---

Incorporate BERT to our task.



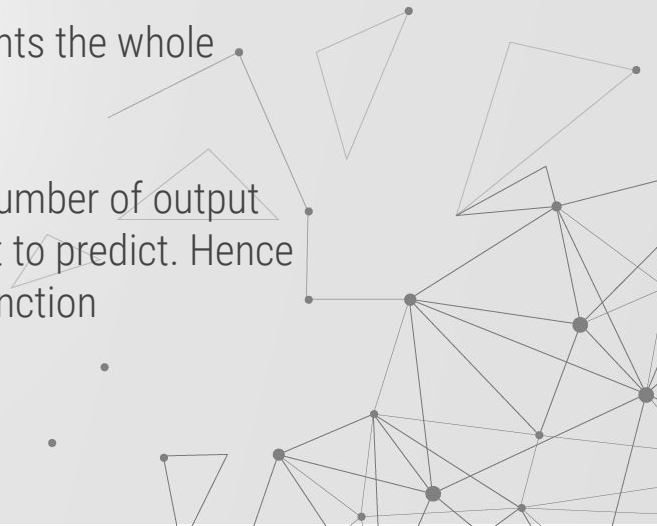
# BERT

- ❖ **BERT** is a neural network architecture that incorporates the encoder layers of the transformers in order to extract contextualized embeddings.
- ❖ The extracted embeddings are extracted in order for the model to excel in the task of masked language modeling and to next sentence prediction
- ❖ Before training BERT model tokenizes the given instances, keeps the first K number of tokens, adds the positional encodings and masks or replaces with another token a percentage of tokens.
- ❖ Also, adds to each instance special tokens that denote its start and end.
- ❖ The masking and replacing are the real power of BERT model.
- ❖ Then the preprocessed instances are passed to the transformer which extracts contextualized embeddings for each token.



# BERT FOR SEQUENCE CLASSIFICATION

- ❖ In order to apply BERT for sentiment analysis we do the following steps:
  - Preprocess each instance as usual
  - Pass it to the neural network in order to extract the contextualized embeddings.
  - Because the representation of the first token represents the whole sequence we consider only that embedding.
  - Above that embedding we apply a Linear layer with number of output neurons equal to the number of classes that we want to predict. Hence the Linear layer should use the Softmax activation function



# 05

## EXPERIMENTAL EVALUATION



# EXPERIMENTS SETUP

- ❖ We evaluated each method on IMDb Reviews dataset which contains 25000 reviews for training and 25000 reviews for testing.
- ❖ Each review will be labeled as positive or negative.
- ❖ For each method we will do different experiments.
- ❖ Each method will be evaluated according to F1-Score and accuracy.



# MACHINE LEARNING EXPERIMENTS

- ❖ In essence, we evaluated each classifier along with each type of features that we extracted.
- ❖ The classifiers that we used are the Naive Bayes and SVM with different regularization parameters.
- ❖ The types of features are TF-IDF, Bag of Words and Word2Vec embeddings of size 300.
- ❖ From the results it is clear that the best model is the SVM with  $C=1.0$  that uses word2vec features which achieves 86% accuracy.

Model	Bag Of Words		TF-IDF		Word2Vec	
	train	test	train	test	train	test
Naive Bayes	0.7520	0.6840	0.8032	0.7876	0.7540	0.7467
SVM with $C = 1.0$	0.9072	0.8520	0.9088	0.8336	0.9111	0.8660
SVM with $C = 10.0$	0.9618	0.8400	0.9874	0.8313	0.9817	0.8486
SVM with $C = 50.0$	0.9893	0.8144	0.9996	0.7851	0.9992	0.8291

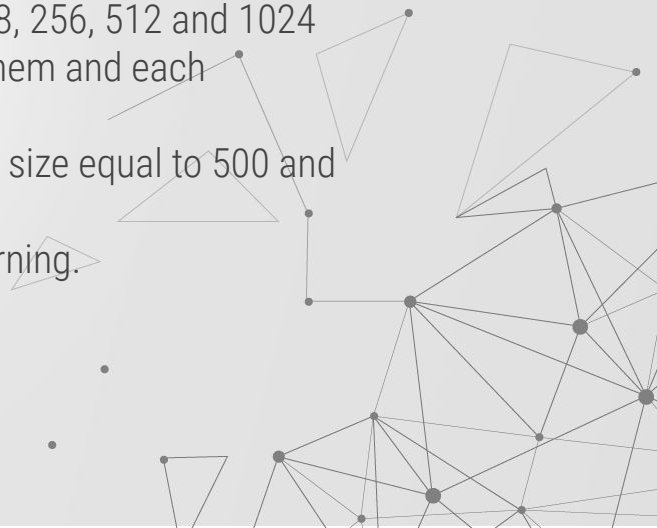
Table 1: Train and Test Accuracies of Machine Learning Sentiment Classifier in IMDB Dataset

Model	Bag Of Words		TF-IDF		Word2Vec	
	train	test	train	test	train	test
Naive Bayes	0.7112	0.6144	0.8029	0.7834	0.7457	0.7357
SVM with $C = 1.0$	0.9083	0.8539	0.9097	0.8348	0.9117	0.8661
SVM with $C = 10.0$	0.9620	0.8412	0.9874	0.8018	0.9816	0.8469
SVM with $C = 50.0$	0.9893	0.8152	0.9996	0.7849	0.9992	0.8271

Table 2: Train and Test  $F_1$ -Scores of Machine Learning Sentiment Classifier in IMDB Dataset

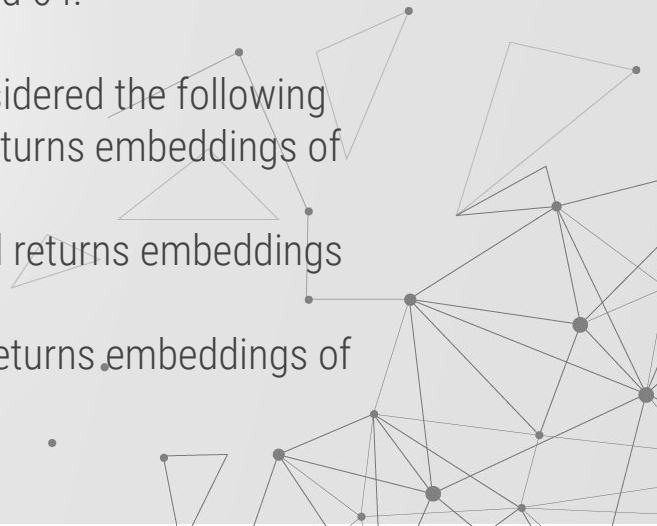
# LSTM RESULTS

- ❖ For LSTM we had to tune the following hyperparameters:
  - The maximum number of tokens that a review will have. This was done in order to reduce the padding due to sentences of outlier lengths. The values we tested was 125, 250, 500, 1000 and 2000.
  - The vector size of the embedding layer. The values we tested were 50, 150, 300, 500 and 1000.
  - The hidden layer size, where we tested the values 64, 128, 256, 512 and 1024
- ❖ We run a Grid Search in order to find the best combination of them and each experiment for 30 epochs and batch size equal to 256.
- ❖ The best combination was the one with max tokens and vector size equal to 500 and hidden layer size equal to 512 with accuracy 86.5%
- ❖ It was not much of an improvement of traditional Machine Learning.



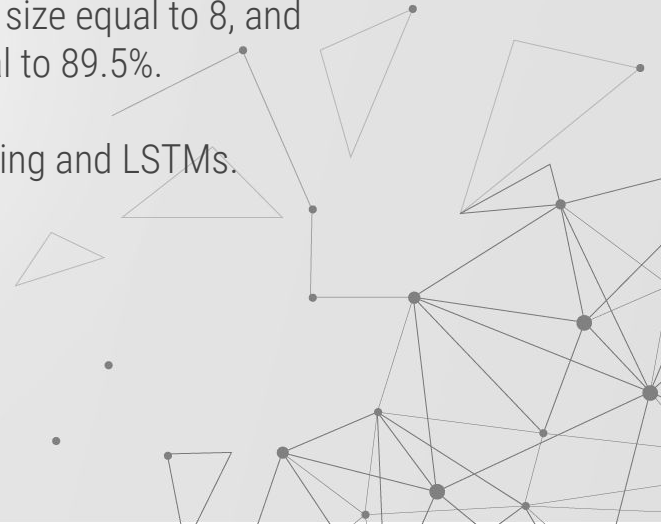
# BERT RESULTS

- ❖ For BERT we had to tune the following hyperparameters:
  - The maximum number of tokens that we will consider for each review. The possible values were 64, 128, 256 and 512 as these are the most common ones for BERT.
  - The batch size. The possible values were 8, 16, 32 and 64.
  - The pretrained BERT model that we will use. We considered the following
    - Bert-small which uses 4 encoding layers and returns embeddings of size 512.
    - Bert-medium which uses 8 encoding layers and returns embeddings of size 512
    - Bert-base which uses 12 encoding layers and returns embeddings of size 768



# BERT RESULTS

- ❖ We run a Grid Search in order to find the best combination of them and each experiment was run for 5 epochs.
- ❖ That number of epochs was enough for BERT to train.
- ❖ The best combination was the one on bert-medium with batch size equal to 8, and maximum tokens equal to 256 which yielded an accuracy equal to 89.5%.
- ❖ It was an improvement of 3-4% from traditional Machine Learning and LSTMs.






# 06

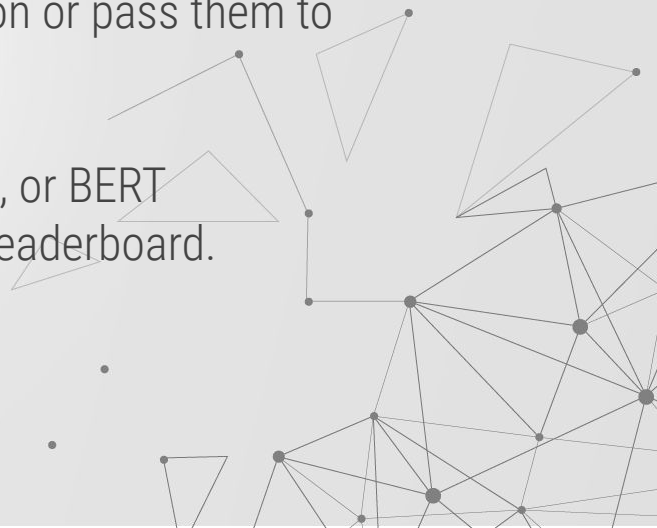
## CONCLUSION & FURTHER WORK

# CONCLUSION

- ❖ We demonstrated three approach to do Sentiment Analysis.
  - ❖ We evaluated their performance and determined that the best one was the BERT one, however we should ask ourselves if its worth due to the resource requirements in comparison to SVMs.
  - ❖ However, with 89.5% accuracy of our best model comparing with PapersWithCode Leaderboard on Sentiment Analysis dataset, our approach is placed 27th which is pretty decent for these simple approaches.
  - ❖ We are only 3% away from top-20, however considering that the best approach achieves 96.1% accuracy we still have a long margin for improvement.
- 

# FURTHER WORK

- ❖ Combine BERT with Machine Learning classifiers by just using BERT as a feature extractor and feed them into a Machine Learning classifier.
- ❖ Apply an ensemble approach where you will pass the predictions of the three classifiers and either vote for the final prediction or pass them to another Machine Learning model.
- ❖ Use another pretrained models like XL-Transformers, or BERT variations like the ones used in top approach in the leaderboard.





# THANKS

Does anyone have any questions?

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

**Please keep this slide for attribution.**