IT3190E – Machine Learning

SENTIMENT ANALYSIS WITH IMDB DATASET

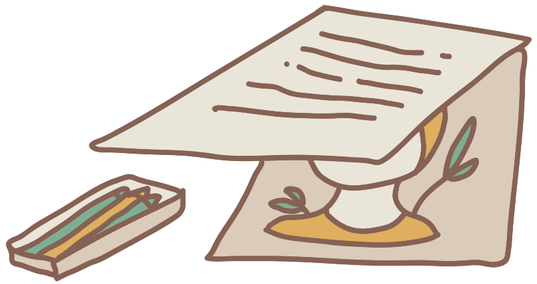
Outline

1. Problem Statement
2. Machine Learning Approach
3. Deep Learning Approach
4. Expanded Problem

Problem Statement

Dataset

IMDb Reviews is a large dataset for binary sentiment classification, consisting of 50,000 highly polar reviews (in English) with an even number of examples for training and testing purposes.
The dataset contains 50000 additional unlabelled data.The number of positive and negative sentiment is equal in the dataset (25000 each)



Example

Negative Review (0):
"After months of hype, we are left with a mess of a story that is mostly just CGI blobs, some light shows and some boring dialogue."

Positive Review (1):
"It was interesting to see how easily things can go really bad that even Thanos Crusade that was the big deal can seem so simple and trivial."

Machine Learning Approach

Data preprocessing

TF - IDF

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF

Term x within document y

$tf_{x,y}$ = frequency of x in y

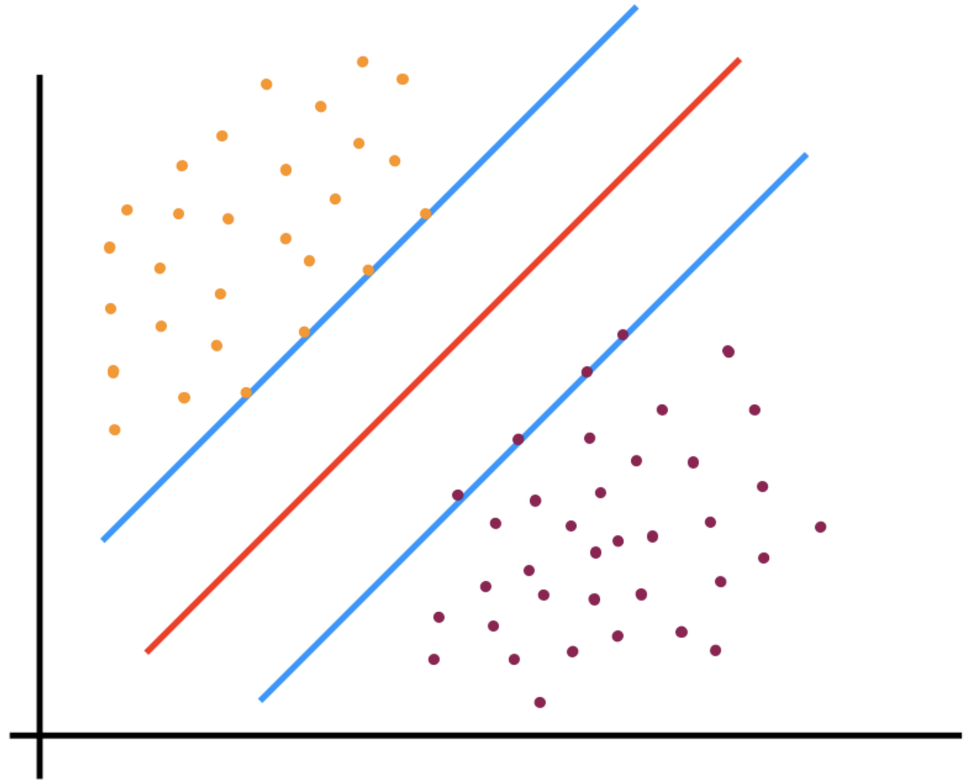
df_x = number of documents containing x

N = total number of documents

Term frequency - Inverse document frequency: is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus

Machine Learning approach

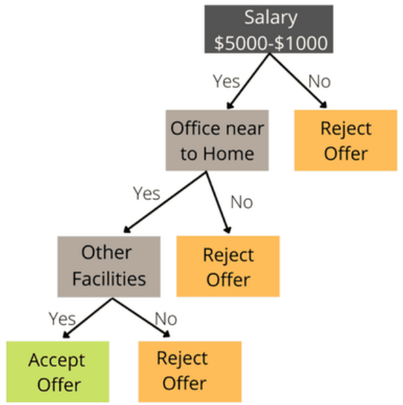
SVM

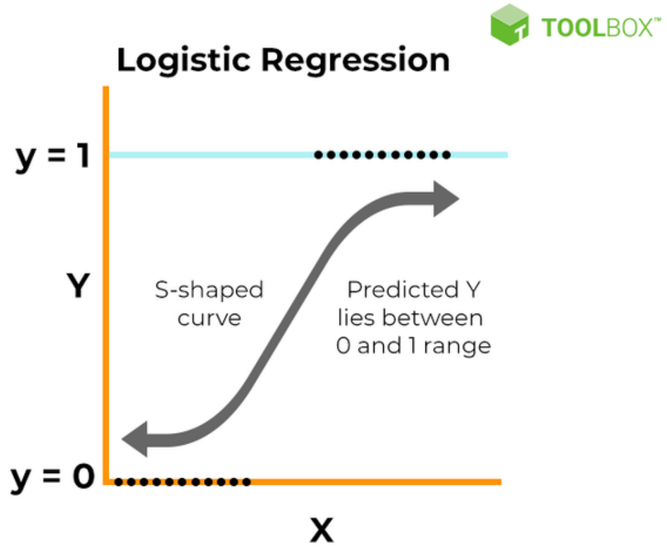


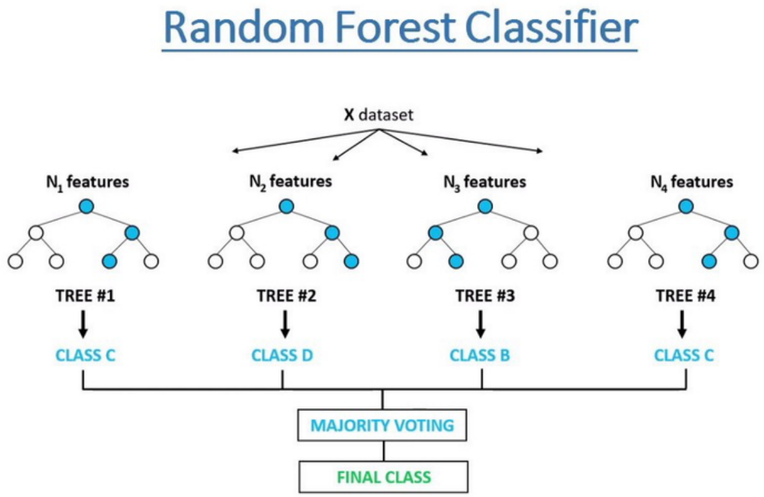
C=0.5
kernel='linear'

Machine Learning approach

Simple Classifier







Decision Tree

criterion = 'entropy'

Logistic regression

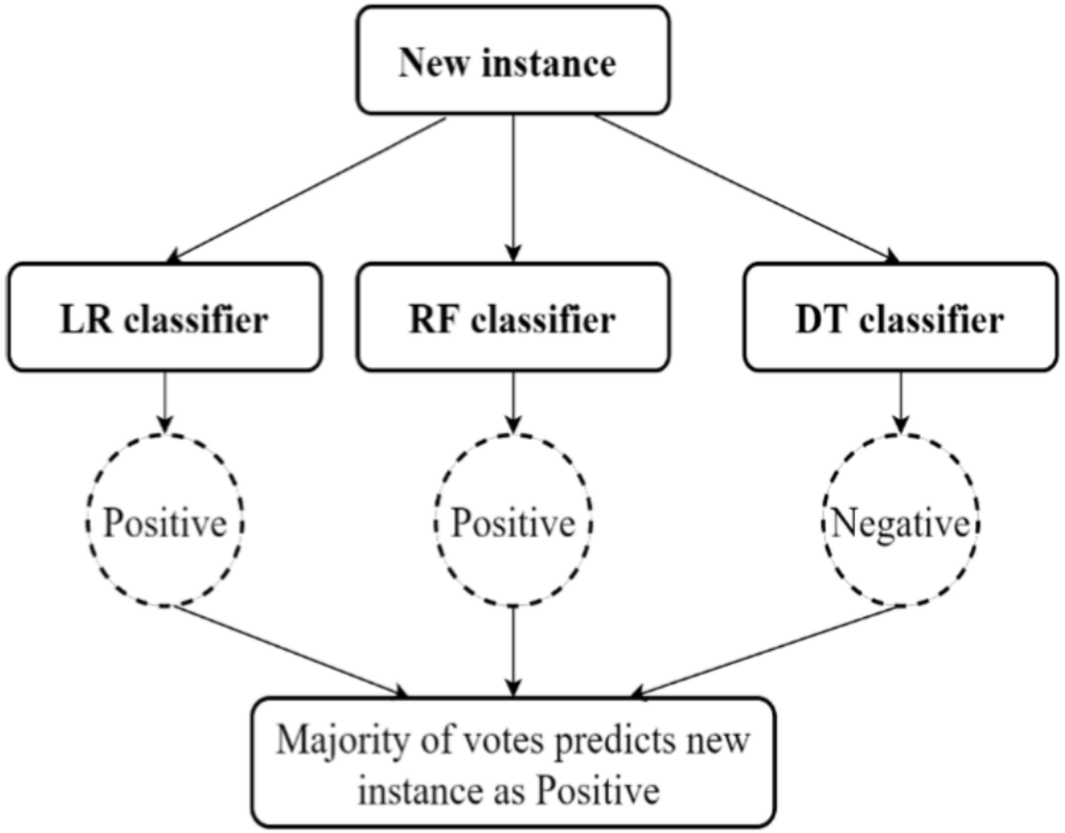
C = 2

Random Forest

criterion = 'entropy'

Machine Learning approach

Voting Classifier



A machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output.

Machine Learning approach

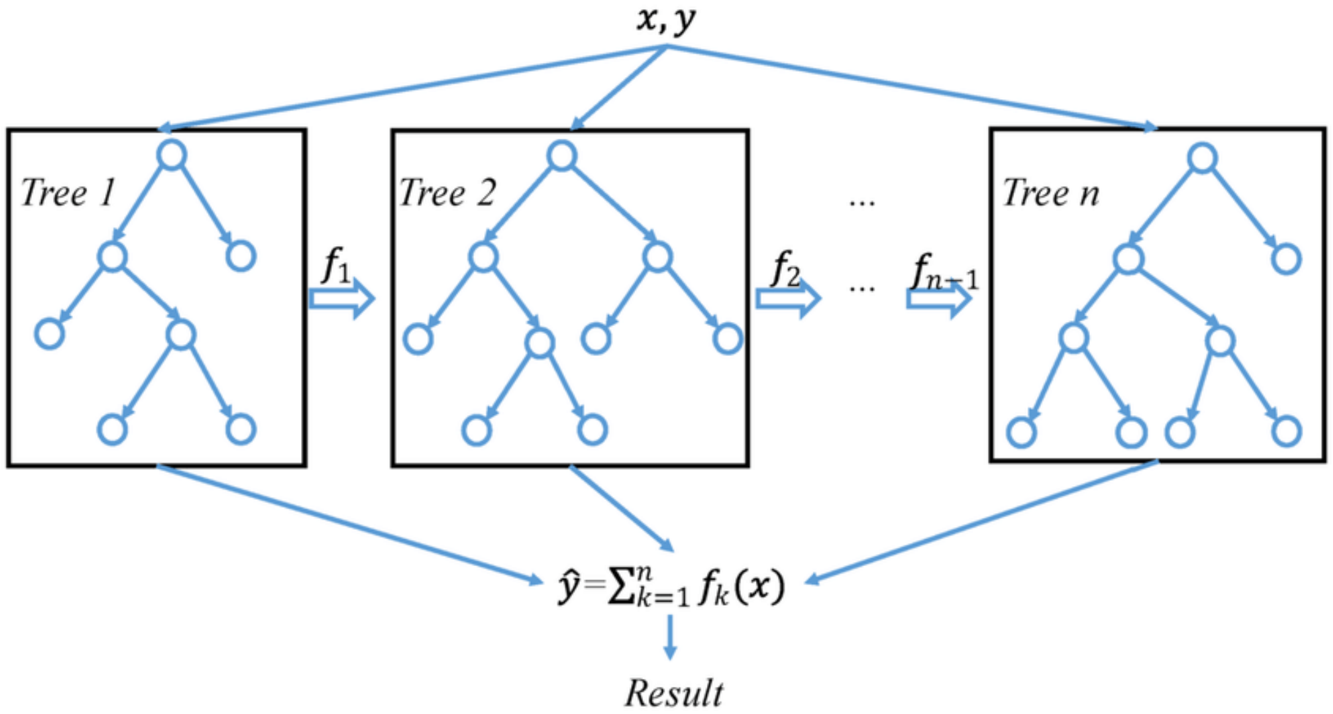
Voting Classifier

Algorithms	Test set accuracy(%)
Logistic Regression	88.32
Random Forest	85.12
Decision Tree	85.20
Hard Voting Classifier	86.93

Table 1: Algorithms in Hard Voting Ensemble Summary

Machine Learning approach

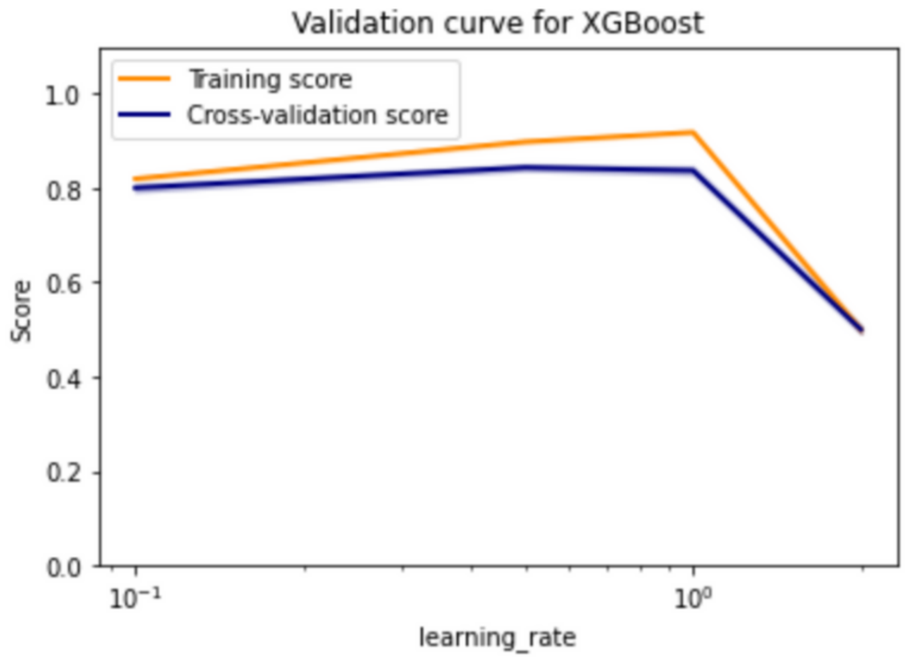
XGBoost

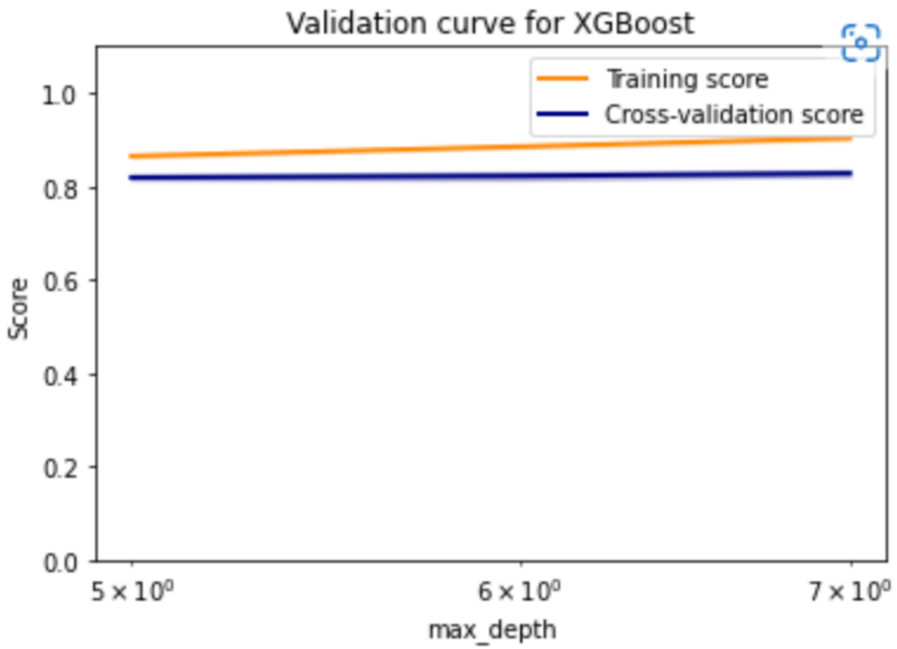


XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm. .

Machine Learning approach

XGBoost

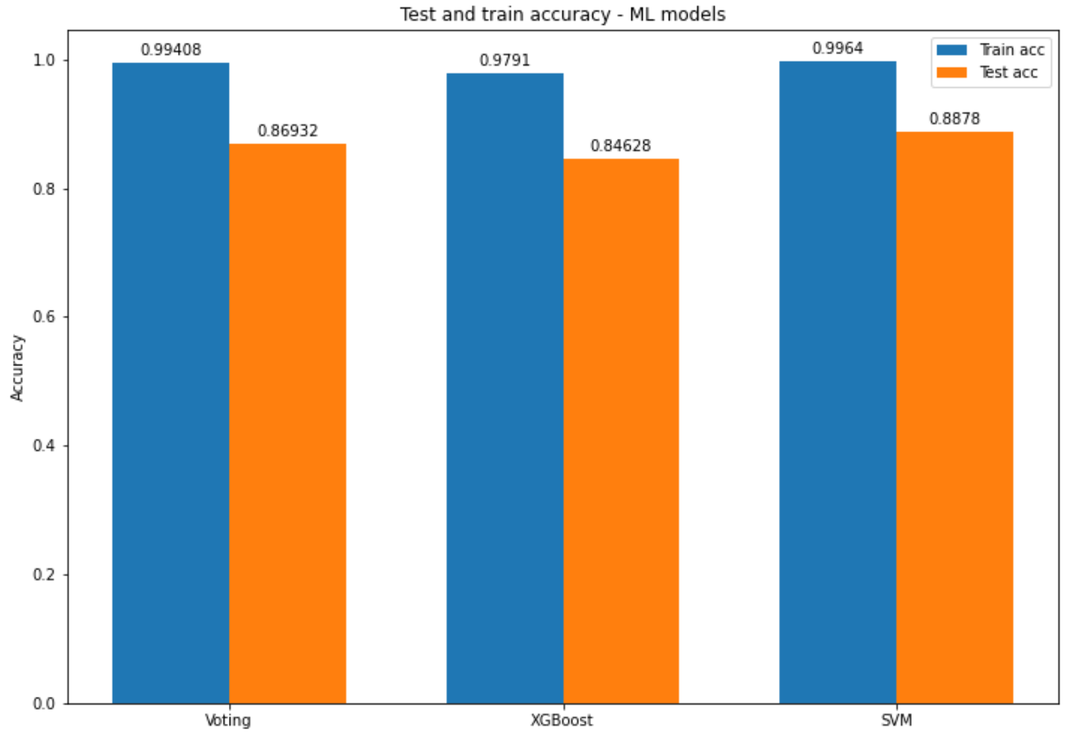




learning_rate=0.5
max_depth = 7

Machine Learning approach

Models evaluation



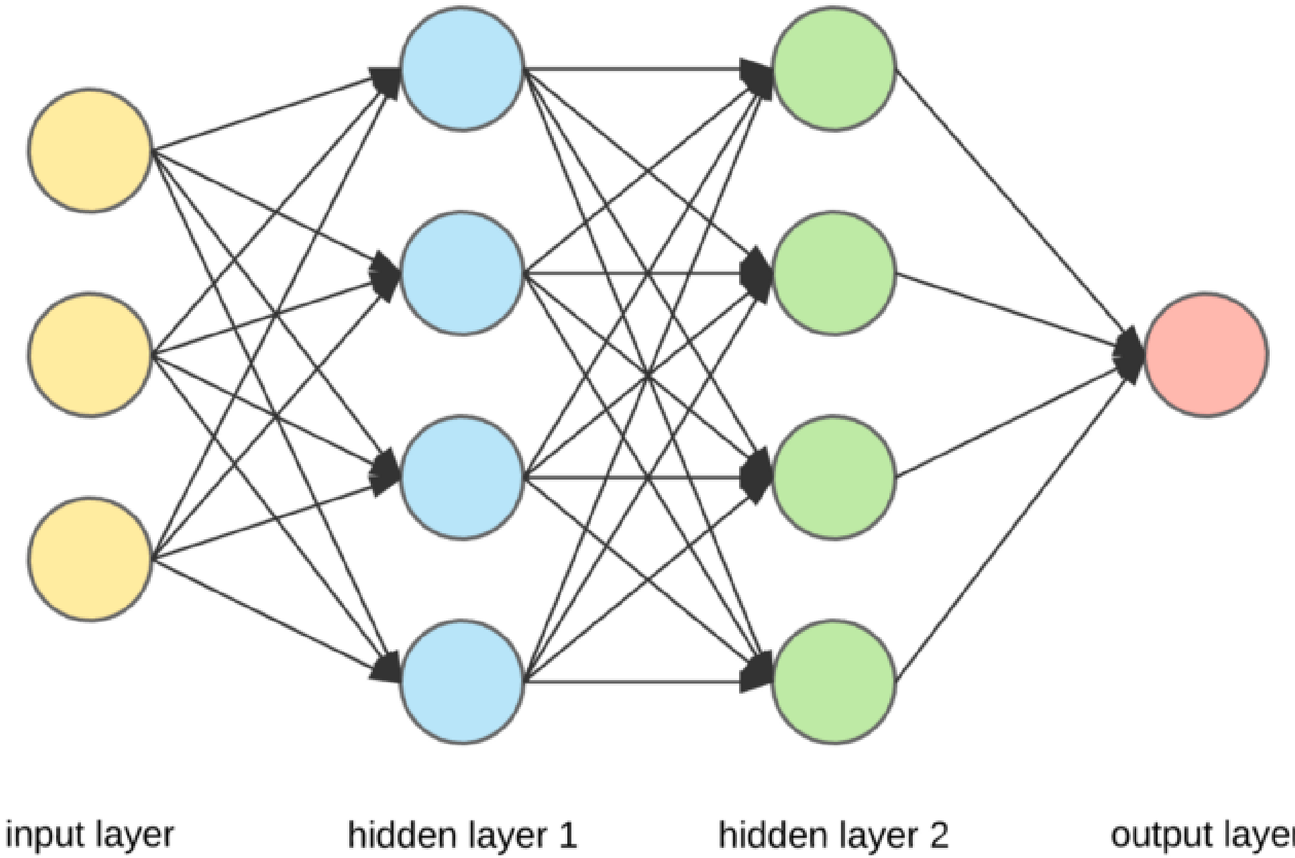
Deep Learning Approach

Neural Network

Neural network as a mathematical function

f: X => y

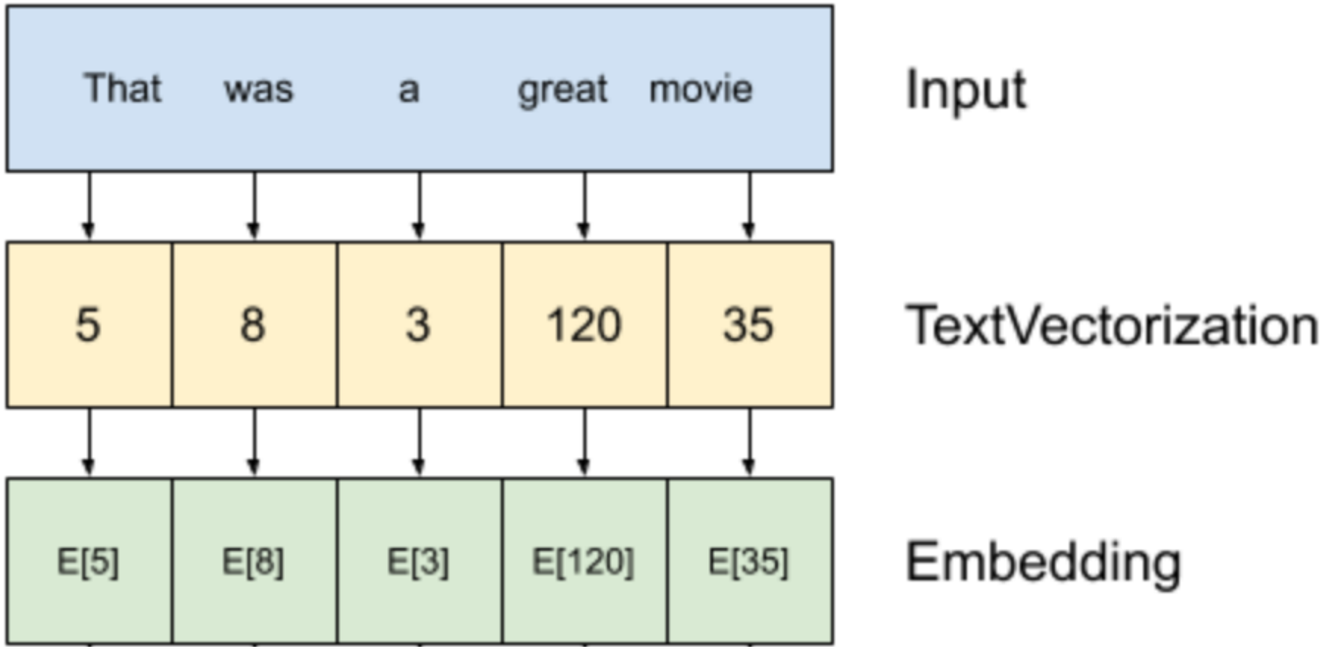
X: words?



Preprocessing: Word Embedding

lower and strip punctuation,
turn word into index

turns positive integers (indexes)
into dense vectors of fixed size

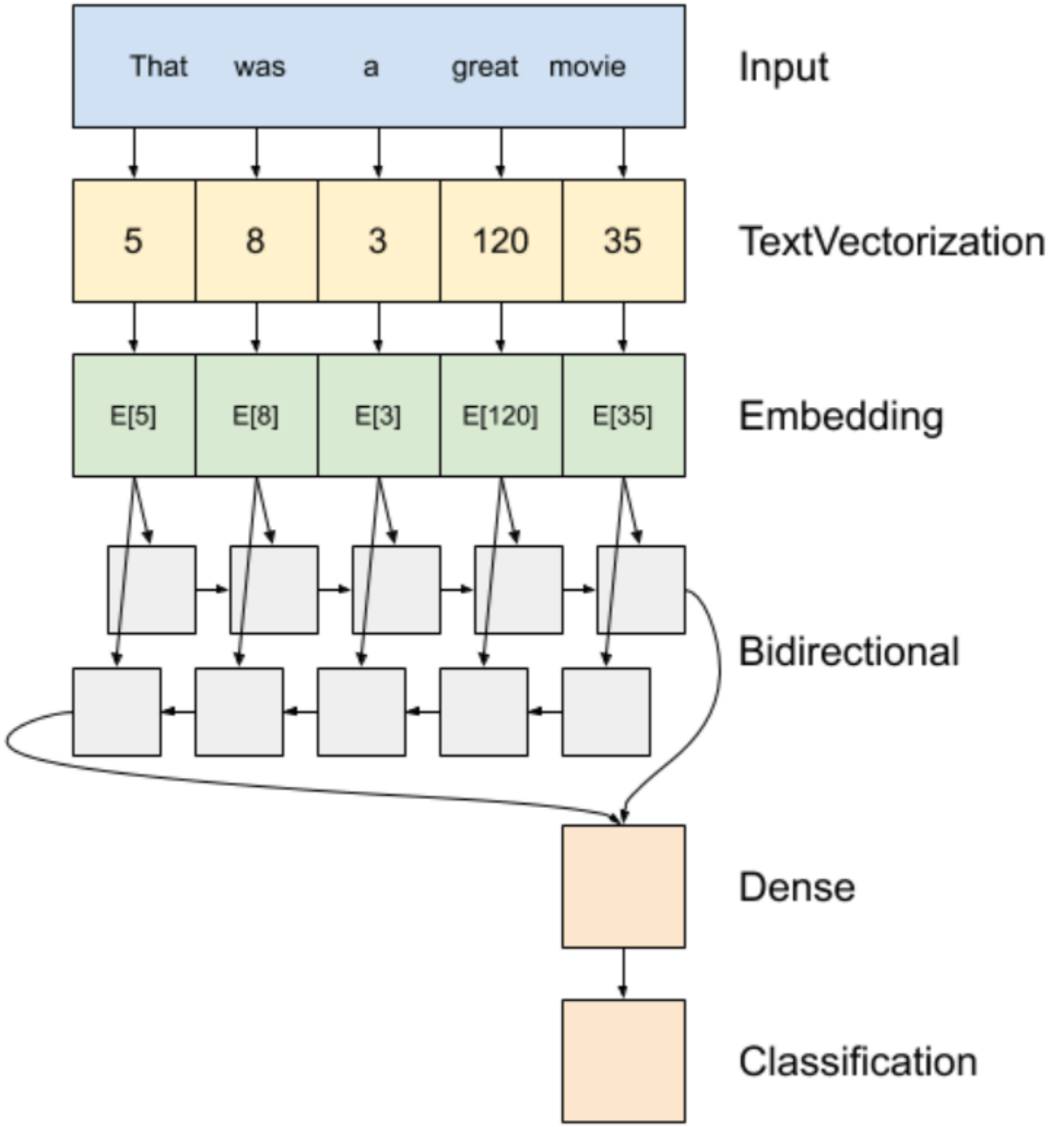


Word embeddings are a type of word representation that allows words that are closer in the vector space are expected to be similar in meaning

Bidirectional LSTM Neural Network

Word embedding

Feature Extraction



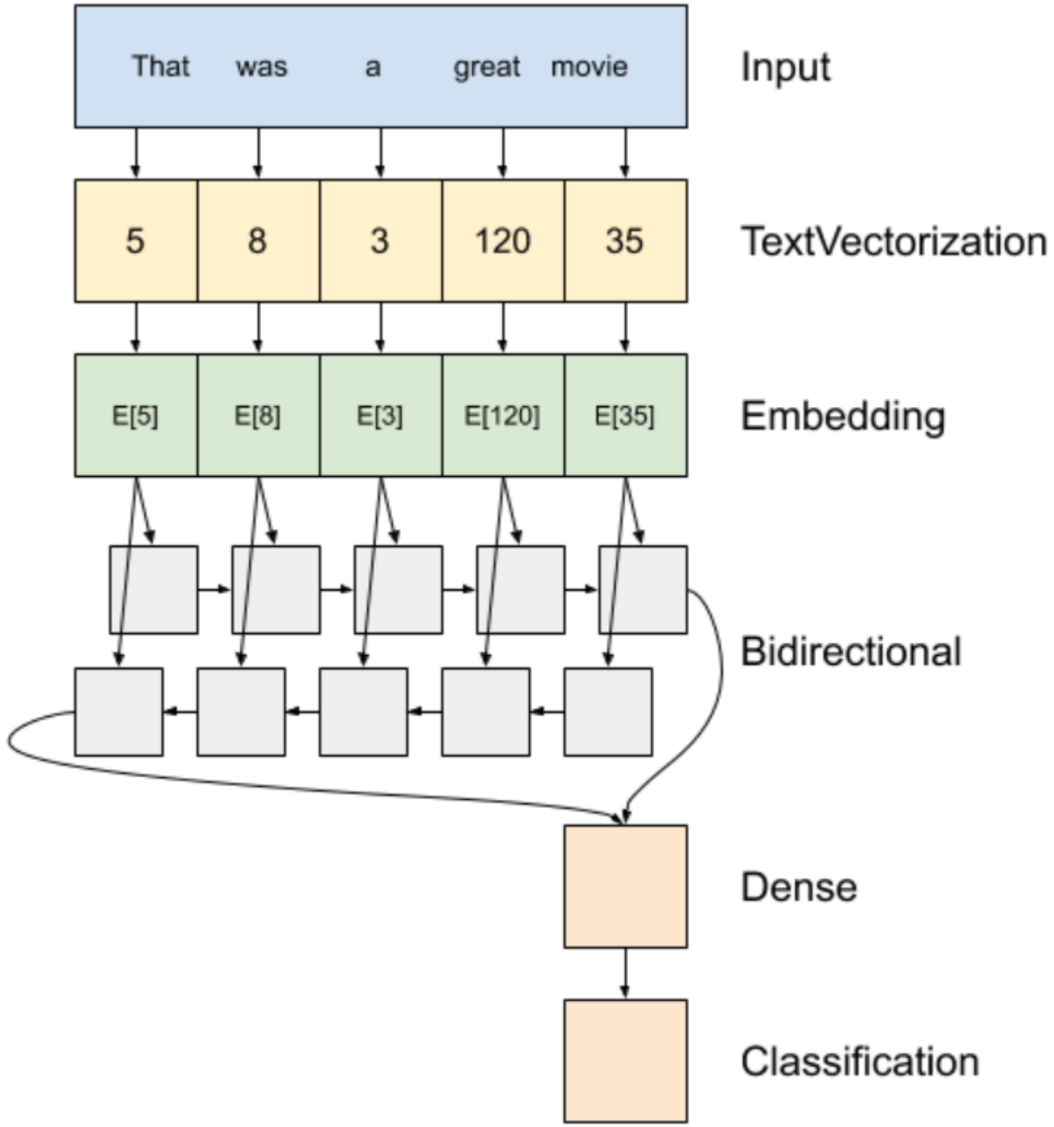
Deep Bidirectional LSTM Neural Network

Deep bidirectional LSTM

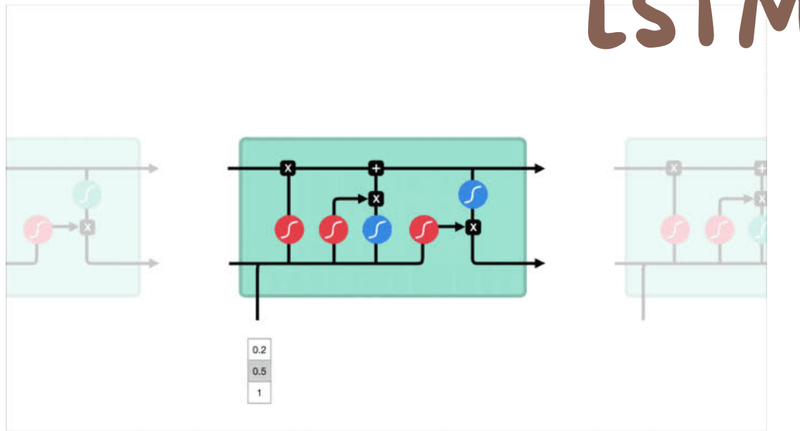
- Similar to bidirectional LSTM.

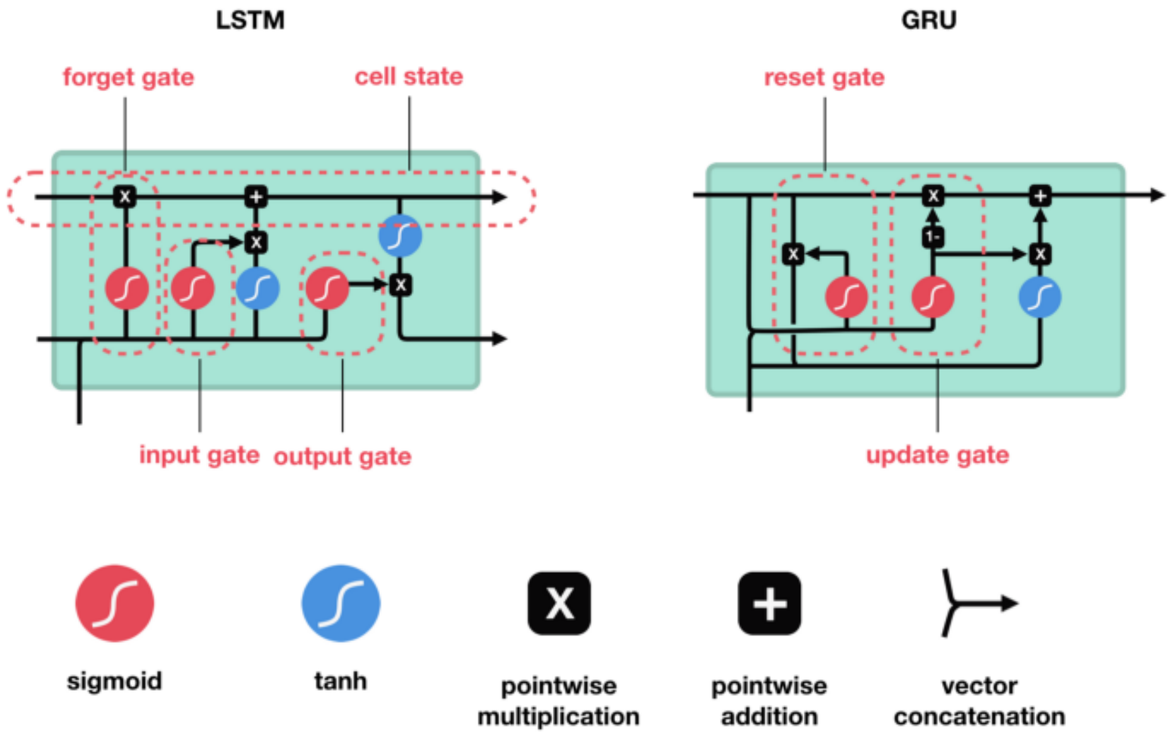
- Difference: multiple layers per time step.

- Higher learning capacity, need larger dataset



LSTM cell replacement: GRU





GRU - Gated recurrent unit

- Newer generation of Recurrent Neural networks

- Similar to an LSTM cell

- Used the hidden state to transfer information

DBNet: Evaluation

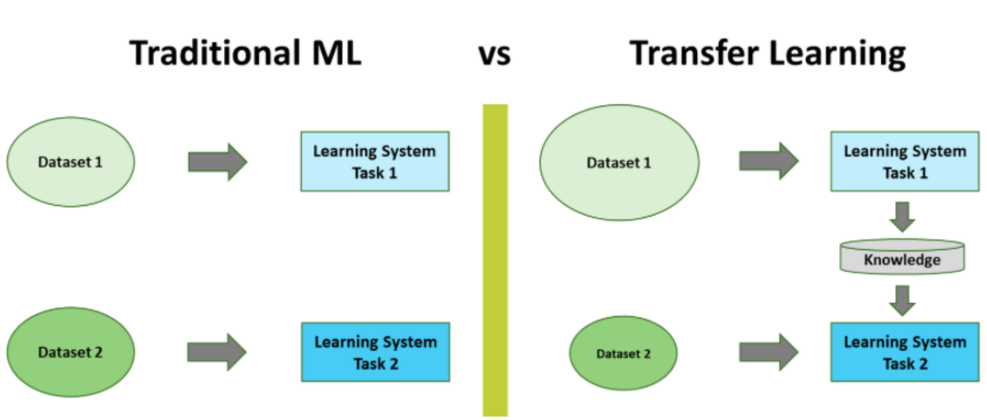
Deep Bidirectional Network Performance

Accuracy(%)	DB-LSTM (4e)	DB-GRU (2e)
Train	0.9790	0.9724
Test	0.8772	**0.8893**

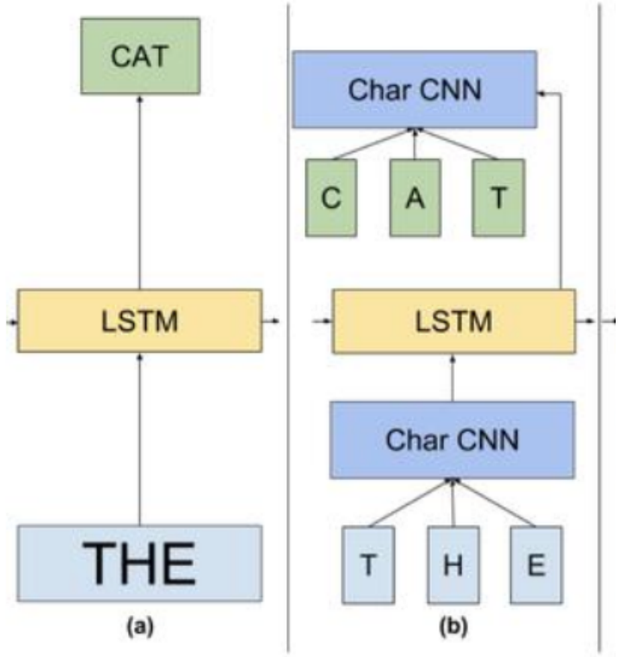
Deep Learning Approach

UMLFiT

Language Modeling

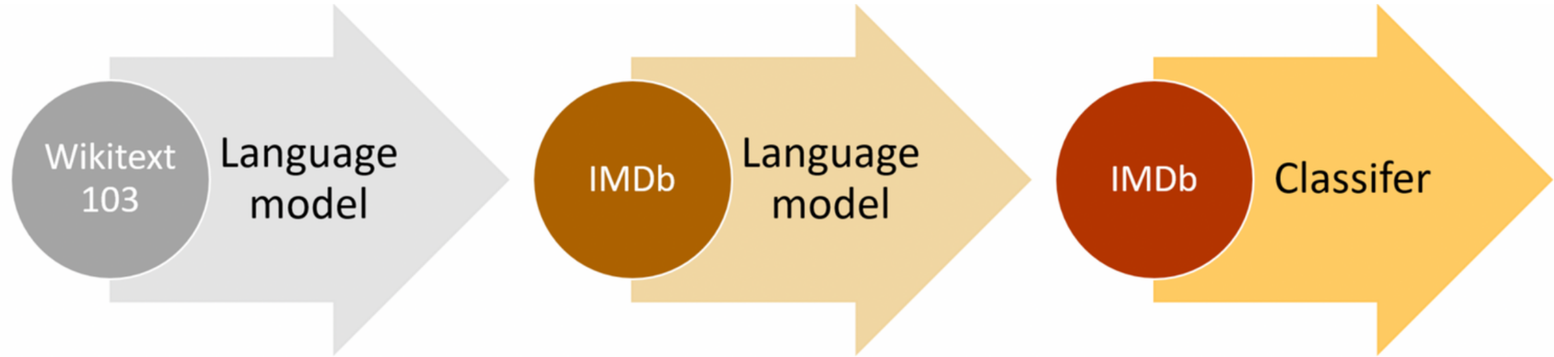


Transfer learning: transfer of knowledge from a related task that has already been learned.



Language modeling task:
- predicting the next word or character in a document
- can be used to train language models that can further be applied to a wide range of NLP tasks

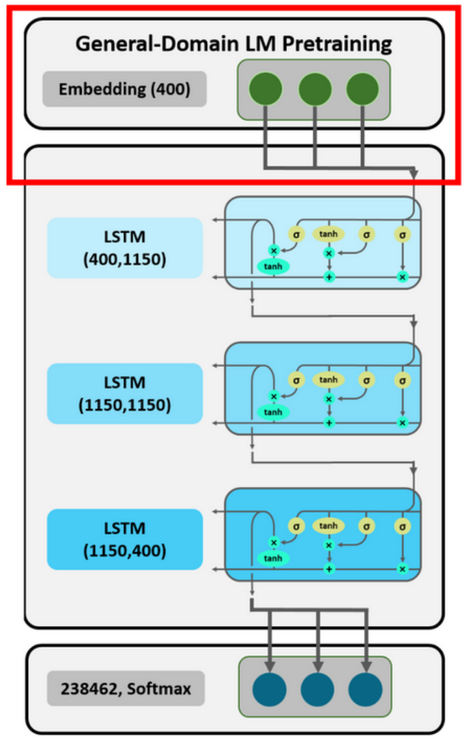
UMLFiT - Universal Language Model Fine-tuning



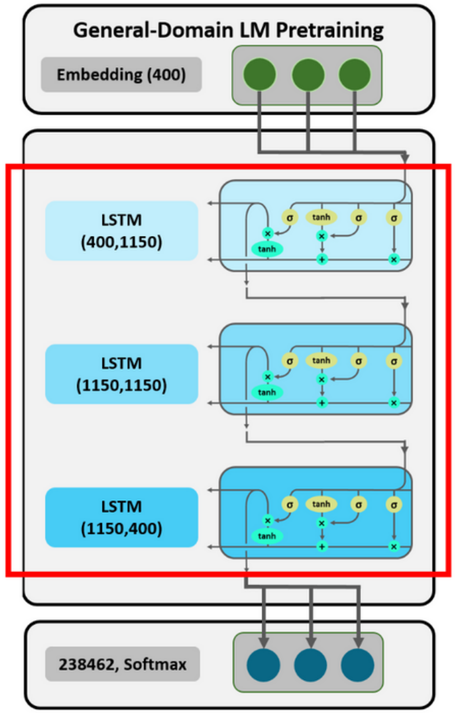
An architecture and transfer learning method, involves a 3-layers AWD-LSTM architecture for its representations.

Three stages:
- General-domain LM pretraining: Training LM in larger corpus
- Target task LM fine-tuning: Fine-tuning LM with target data
- Target task classifier fine-tuning: Fine-tuning classifier with add linear block

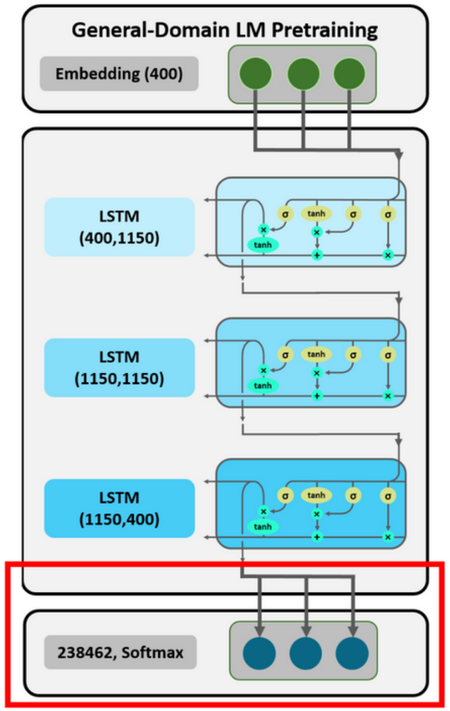
General-domain LM pretraining



Matching every encoded token to its embedding vector executed via one hot

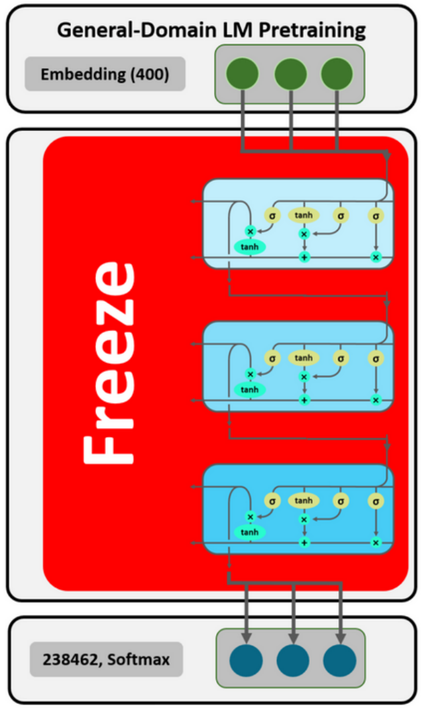


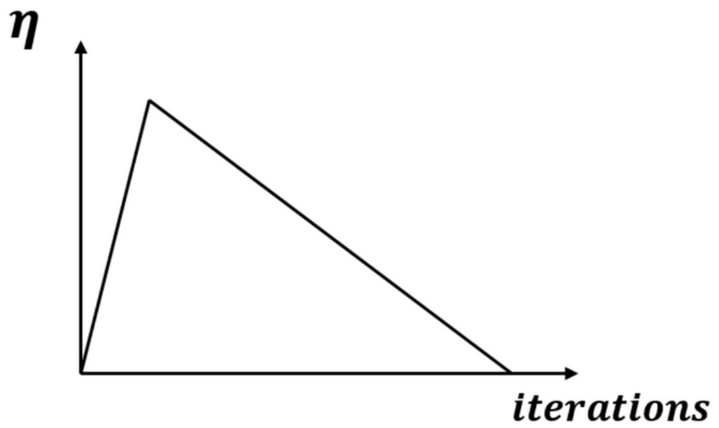
3 stacked LSTM layers exclude dropouts (DropConnect) & optimize by Average-SGD



Softmax function transforms all values in the decoded tensor into probabilities, evaluated by perflexity

Target Task LM Fine-tuning






```
# set discriminative learning rate
lr  = 1e-2
lrm = 2.6
lrs = np.array([lr/(lrm**3), lr/(lrm**2), lr/lrm, lr])


# fit model with unfrozen lstm layers until it overfits
learner.fit(lrs, 1, wds = wd, use_clr = (20,10), cycle_len = 3)
```

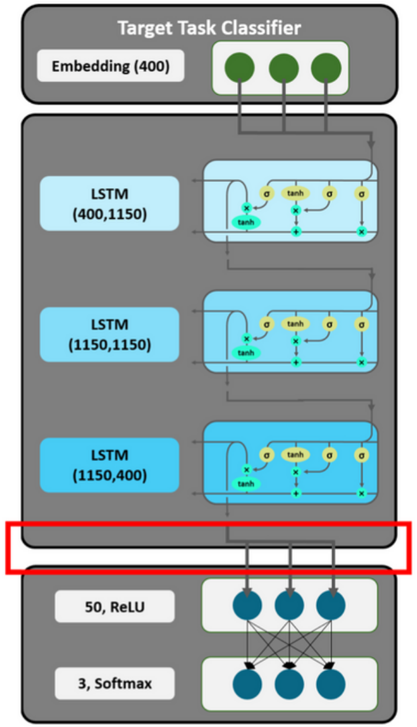
```
HBox(children=(IntProgress(value=0, description='Epoch', max=3,
tion_width='initial…
epoch      trn_loss   val_loss   accuracy
    0      4.336937   3.914854   0.274203
    1      3.92364    3.78316    0.288591
    2      3.69587    3.764121   0.291678
[array([3.76412]), 0.2916783712553174]
```

Freeze weight LSTM layers and training rest of model 1 epoch - avoid catastrophic forgetting, then unfreeze all layers for fine-tuning

Adjusted LR, init smallest, increase to quickly converge to suitable region parameters, then decrease for precise fine-tune weights
Optimal LR chose smaller point when loss start increase

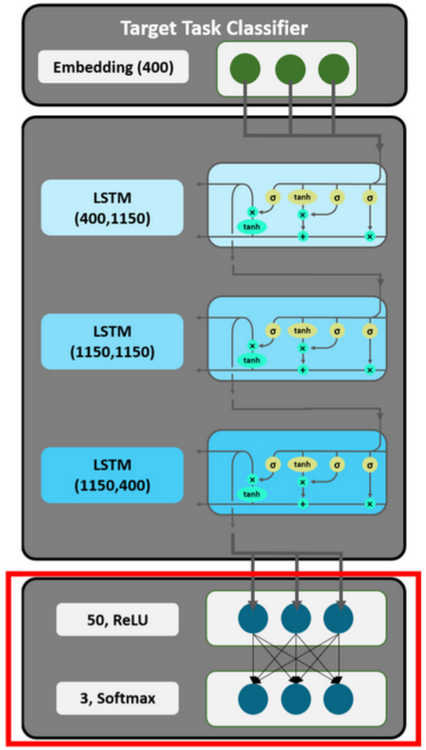
Different layers has different LR to capture different informations

Target Task Fine-tuning



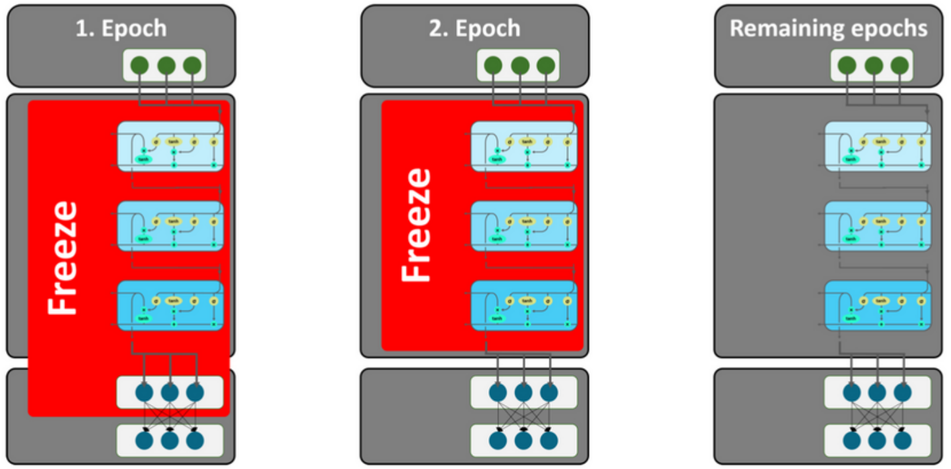
Concat Pooling

Avoid catastrophic forgetting, last hidden state concatenated with max-pooled and mean - pooled representation of the hidden states



Linear Decoder

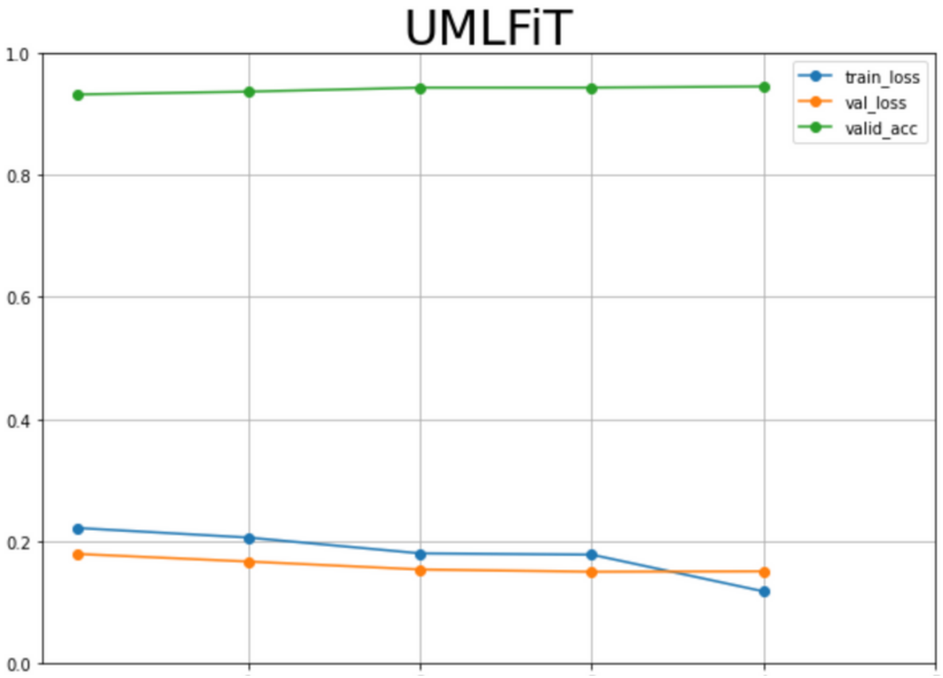
Cut-off Softmax of LM by RELU and Sigmoid on top LSTM, using batch normalization and dropouts

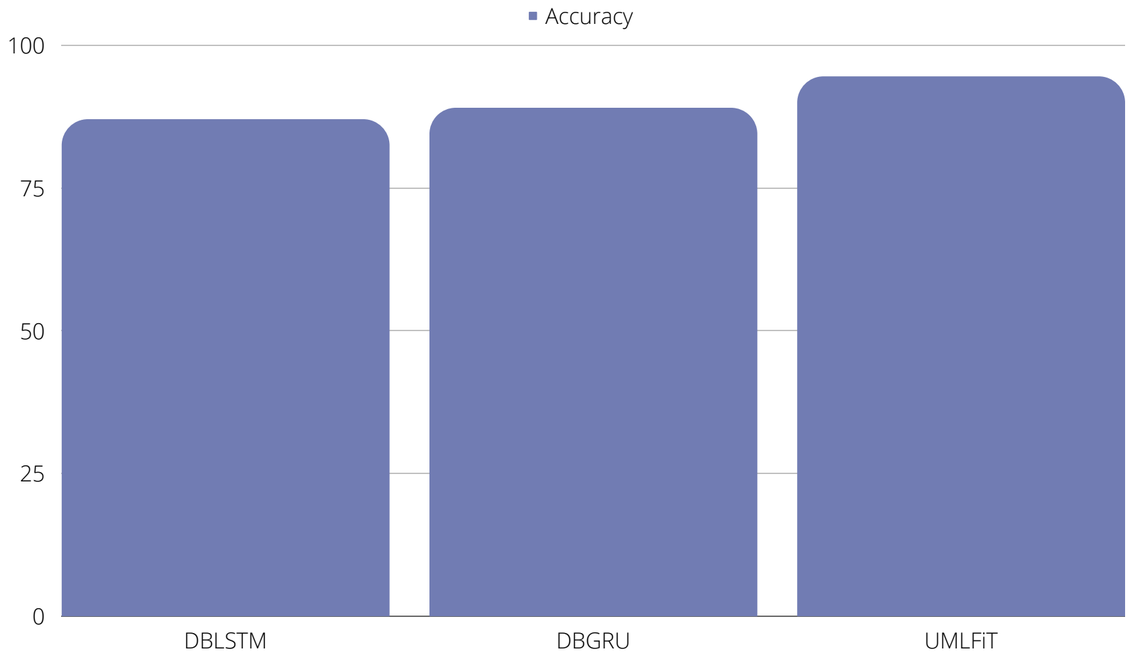


Gradual Freezing

Linear layers init random weight risk catastrophic forgetting, so it fine-tuned first, after that stack 3 LSTM and finally all model

Evaluation

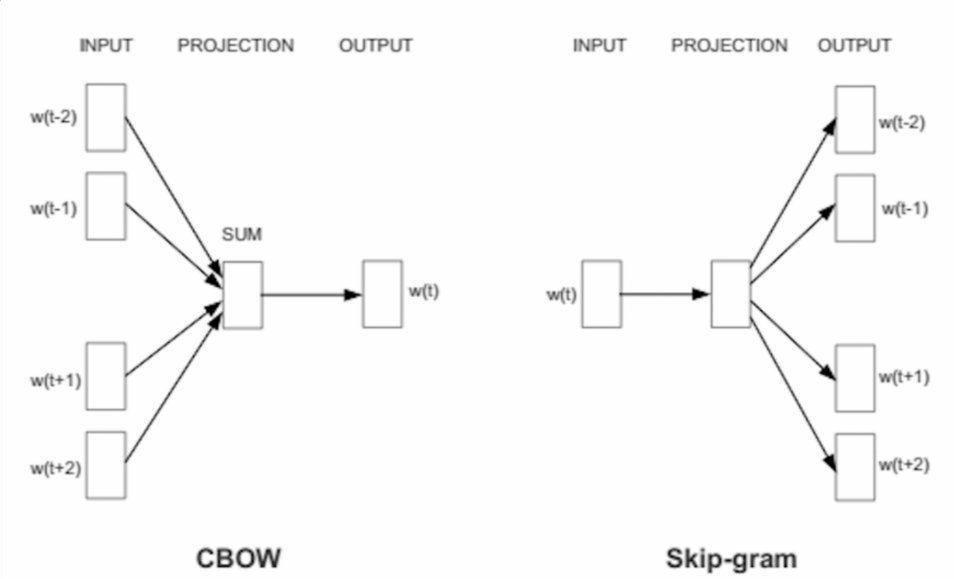


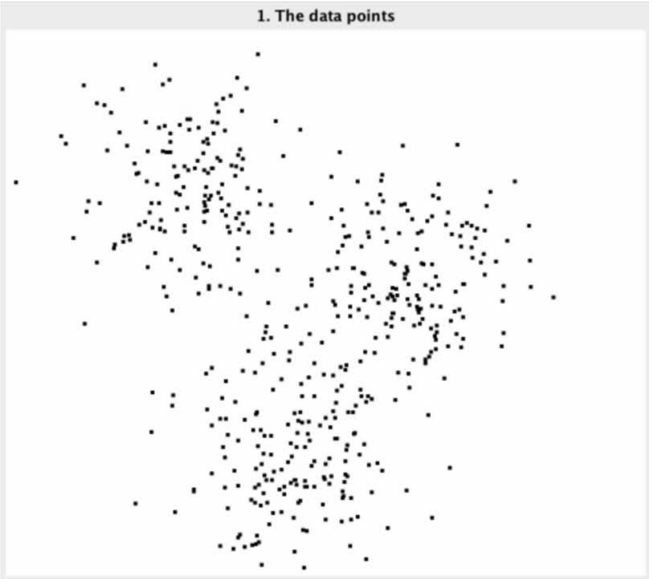


UMLFiT collected acc 94.5% after **5 epoches** train
(3x unfrezze transfer layers)

Unsupervised Learning

Pre-trained Word2Vec x Mini-batch K-means

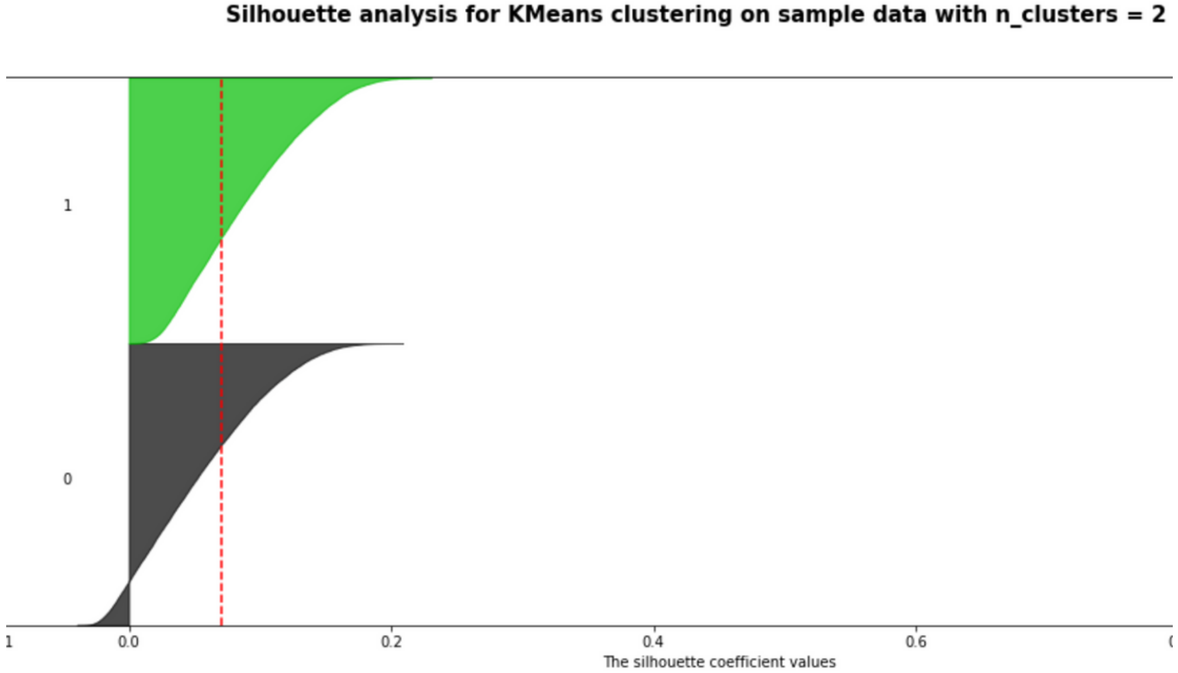




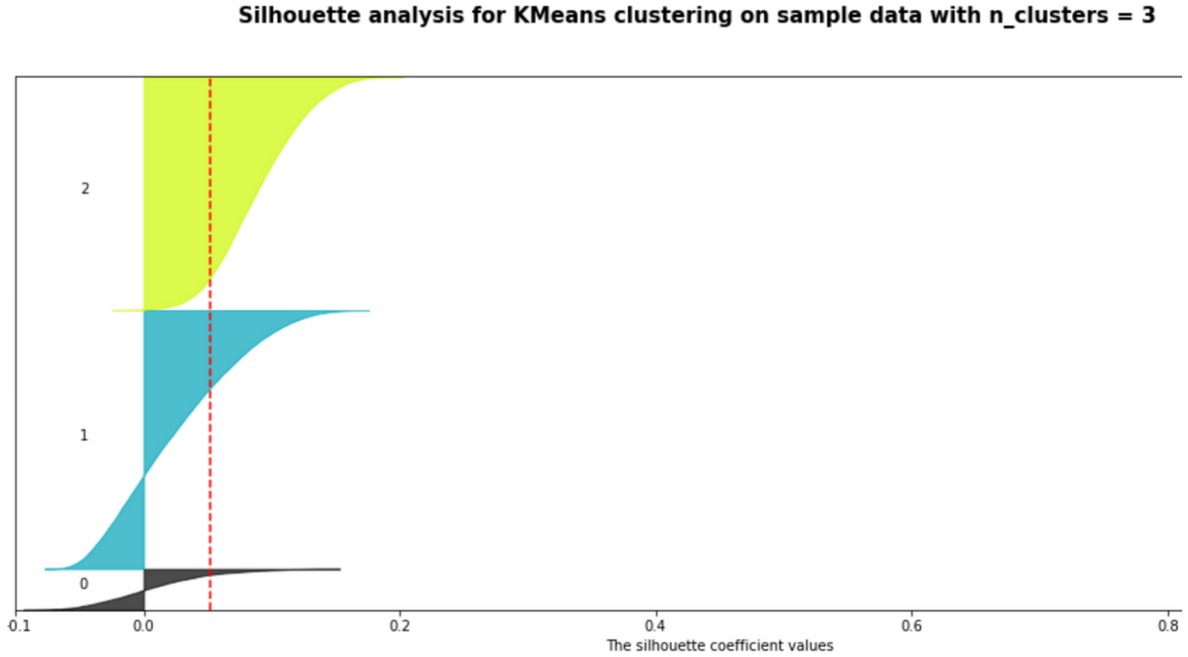
Generating embedding vectors using pre-trained model Word2Vec

Apply Mini-batch K-means for clustering

Evaluation clustering



2 Clusters with hypothesis data has positive reviews and negative reviews



3 Clusters with hypothesis data has positive reviews, negative reviews and neutral reviews

Thank You!

Do you have any question?