Neural Network-based Language Models Implementation on Goodreads dataset

Babaei Elham, Moret Chiara, Moustafa Fatma

Deep Learning, 2022

Word embeddings

Word embeddings

- Word --> numerical representation
- Not just read or process word
- Build relationship between each word in sentence or document
- Capture:

✓Context

✓Semantic

✓Syntactic properties

✓Similarities

Types of words embeddings

Frequency based

- Word count
- TF-IDF: Term Frequency- Inverse Document Frequency
- Co-occurrence

Prediction based

- CBOW: Continuous Bag Of Words
- Skip-gram

word2vec

word2vec

- Use vector = list of numbers to represent words
- Capture semantic relations
- Use cosine similarity metric
- Similar words have similar vector representations that is captured by the cosine similarity
- Find if a pair of words have a similar connection that another pair of words have
- Syntactic and grammarly based relations

Models used in word2vec

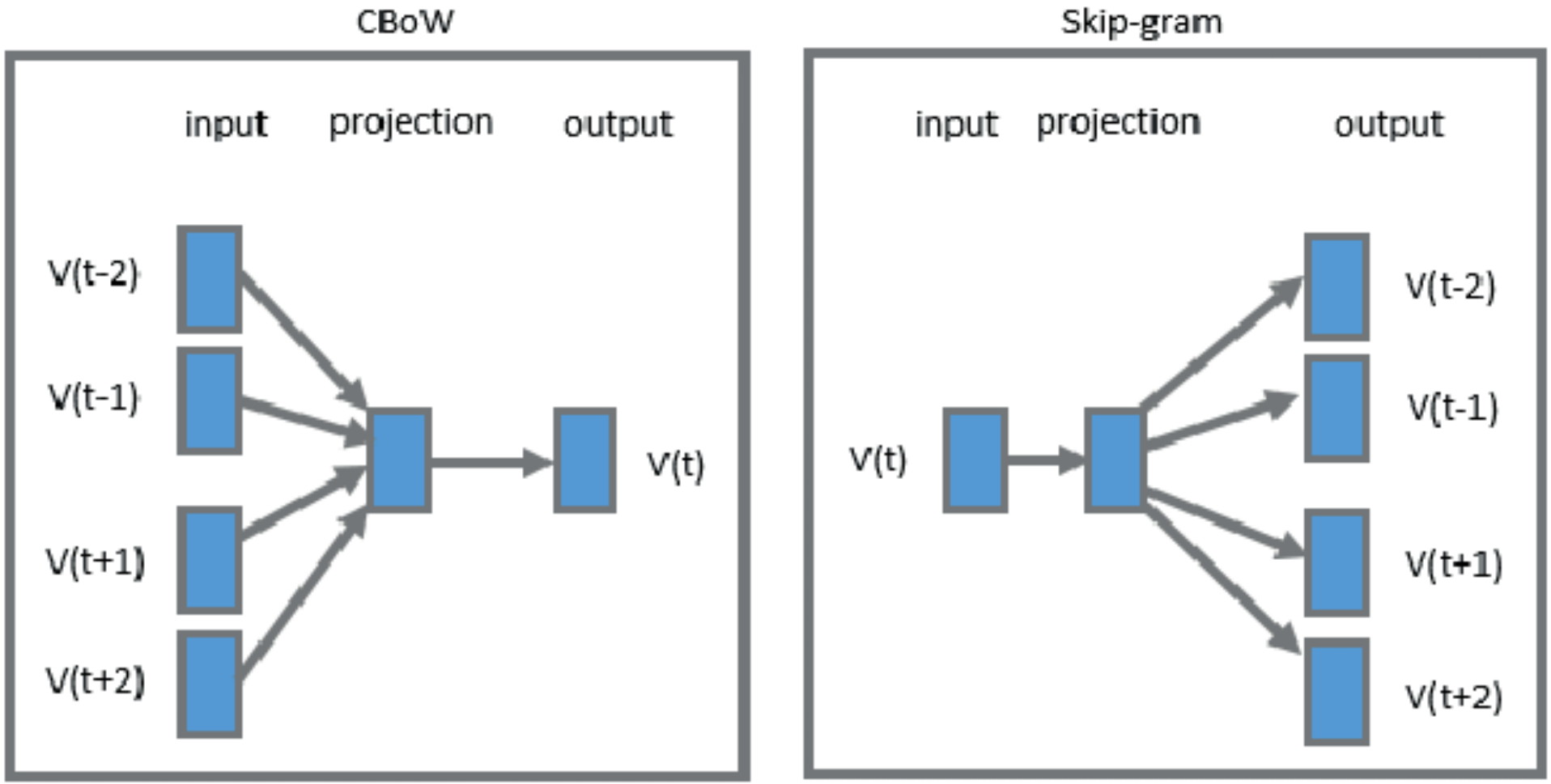
CBOW

- Objective: predict target word
- Input n words before and n words after target
- Target is supposedly closely related to the context of the input words
- Fast and less complex
- Finds better numerical representation of frequent words

Skip-gram

- Objective: predict closely related context of a word
- Input: one word
- Prediction of surroundings of a word
- Represents rare words more efficiently
- More complex than CBOW

Word2vec architecture



GloVe

Global Vector for word representation

- Unsupervised learning technique
- Extension of word2vec
- Training performed on aggregated global word to word co-occurrence statistics from a corpus
- Word-word co-occurrence: A word that is likely to occur with another word more than other words.

Comparison between word2vec and GloVe

word2vec

- Capture local context
- Use neighboring words

GloVe

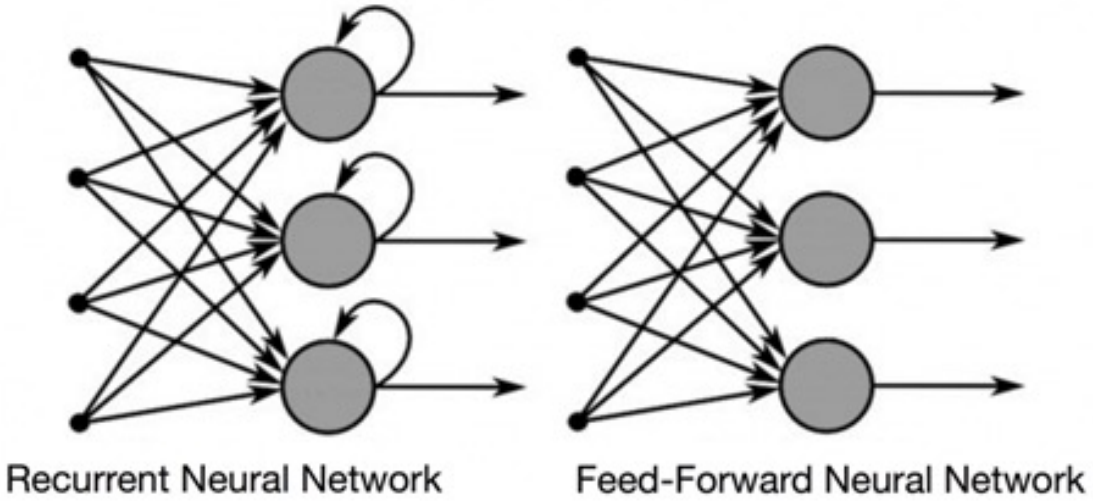
- Consider entire corpus
- Create a large matrix capturing co-occurrence of words within corpus

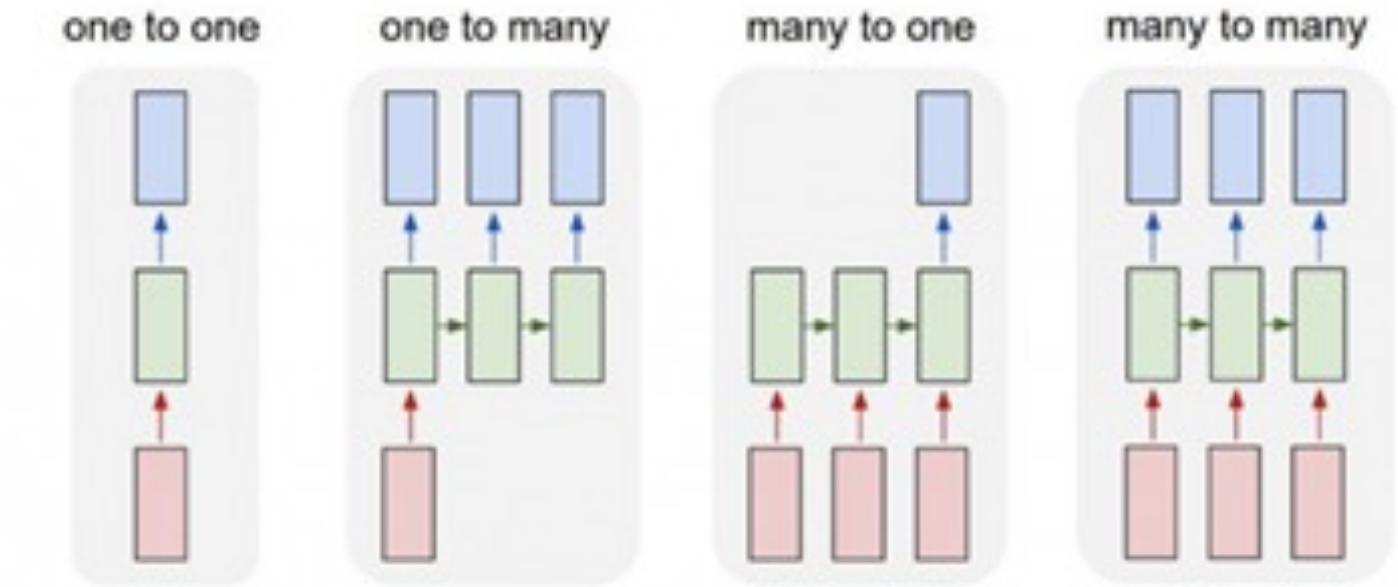
Disadvantage: same vector for the same word regardless of their meaning in a sentence

RNN models

Introduction

- Internal memory
- highly used with sequential data like text, speech, time series, audio, video and more
- RNN the data cycles in a loop. Input coming from previous step are considered when calculating current input.





Applications:

1-1: Binary classification

1-many: Text generation

many-1: Sentiment classification

Many-many: Machine translation

RNN Issues

- Exploding gradients:
  - ✓Truncation
  - ✓Clipping gradient

- Vanishing gradients:
  - ✓Change architecture of RNN by use of gated cells: LSTM, GRU , etc...

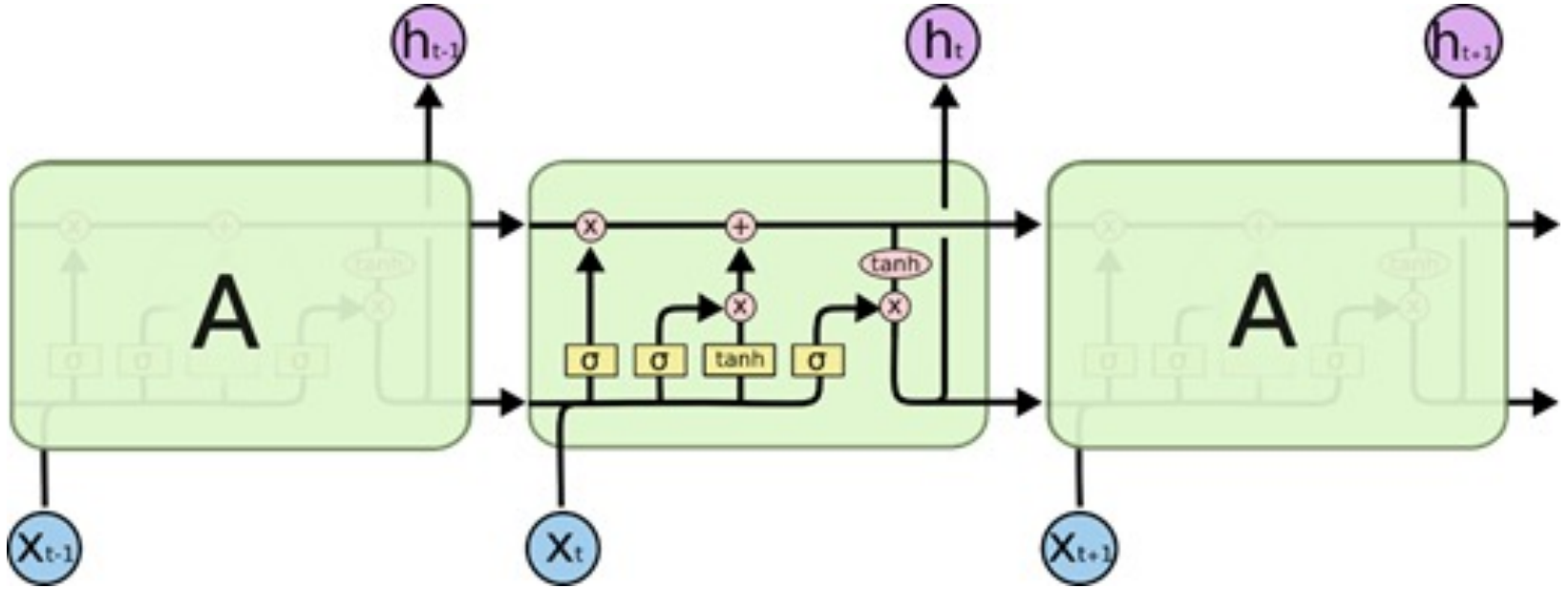
- Encoding bottleneck, slow and no parallelization, not long memory
  - ✓Solution: ATTENTION !!!

LSTM models

General Architecture of LSTM

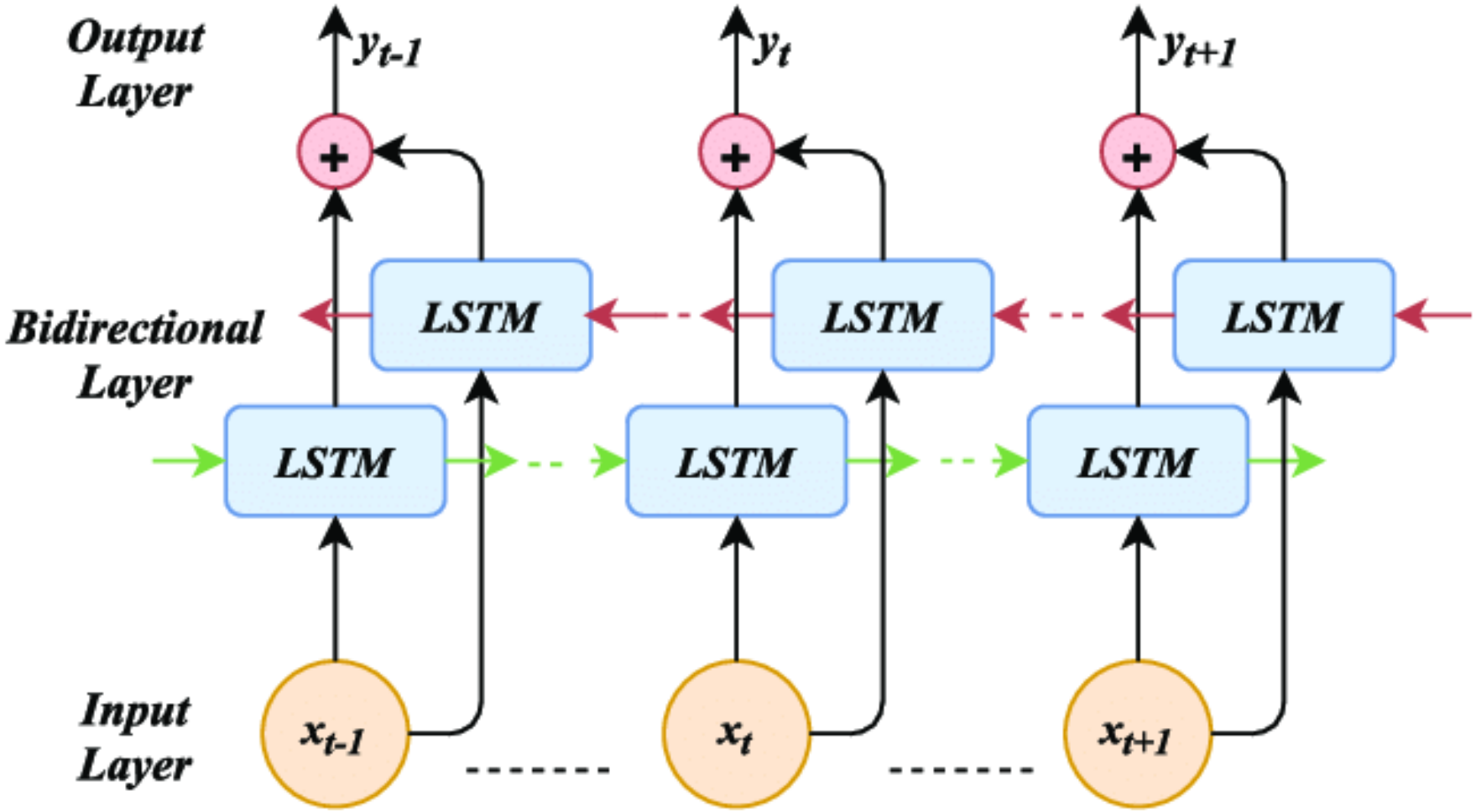
- There are three gates in LSTM:

1. Input gate: determine whether or not to read data into cell
2. Forget gate: delete the unimportant information and resetting cell content
3. Output gate: let it impact the output at the current timestep



Bi-directional LSTM

General Architecture



Bi-directional LSTM

- Input flow in both directions
- Every component of an input sequence has information from both the past and present.

- BiLSTM can produce a more meaningful output, combining LSTM layers from both directions. BiLSTM will have a different output for every component (word) of the sequence (sentence)

- The entire context (essence) of a sentence is captured
- This kind of network can be used in text classification, speech recognition and forecasting models.

ELMo

ELMo: Embeddings from Language Models

- Fixed embedding for each word --> Each token is assigned a representation that is a function of the entire input sentence.
- Vectors are derived from a bidirectional language model (biLM) - bidirectional LSTM
- The same word can have different word vectors under different contexts supporting polysemy.

ELMo representations

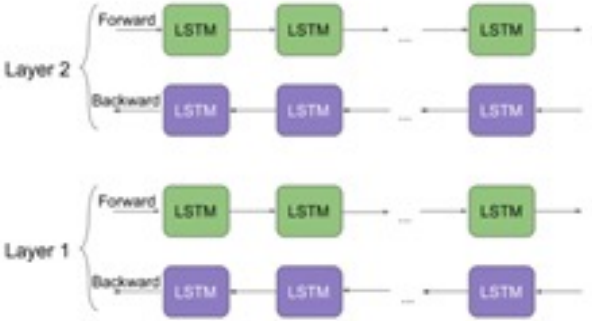
- **Contextual**: The representation for each word depends on the entire context in which it is used.

- **Deep**: The word representations combine all layers of a deep pre-trained neural network. A linear combination of vectors stacked above each input layer is learned instead of just using top LSTM layer.

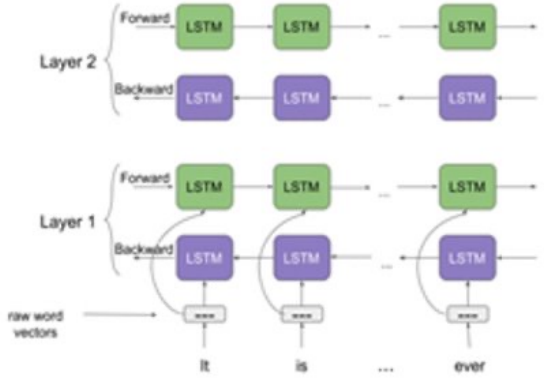
- **Character based**: ELMo representations are purely character based, allowing the network to use morphological clues to form robust representations for out-of-vocabulary tokens unseen in training.

Architecture of ELMo:

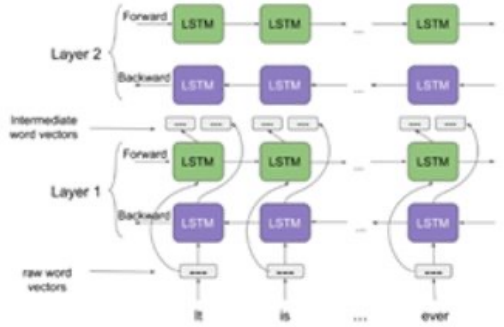
- The forward pass contains information about a certain word and the context (other words) before that word
- The backward pass contains information about the word and the context after it



- These raw word vectors act as inputs to the first layer of biLM

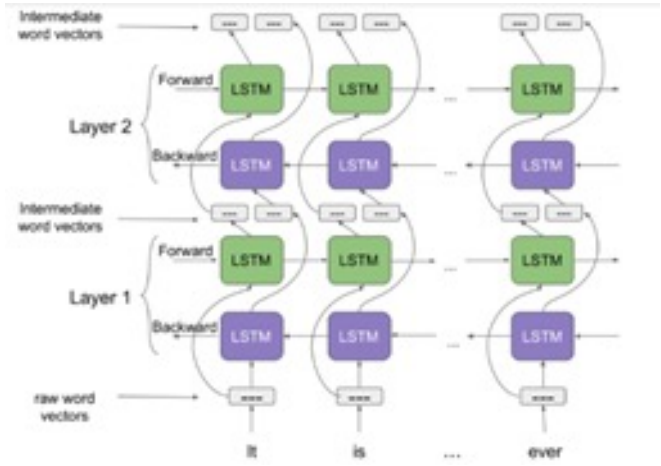


- This pair of information, from the forward and backward pass, forms the intermediate word vectors



Architecture of ELMo:

- These intermediate word vectors are fed into the next layer of biLM



- The final representation (ELMo) is the weighted sum of the raw word vectors and the 2 intermediate word vectors

*BERT models

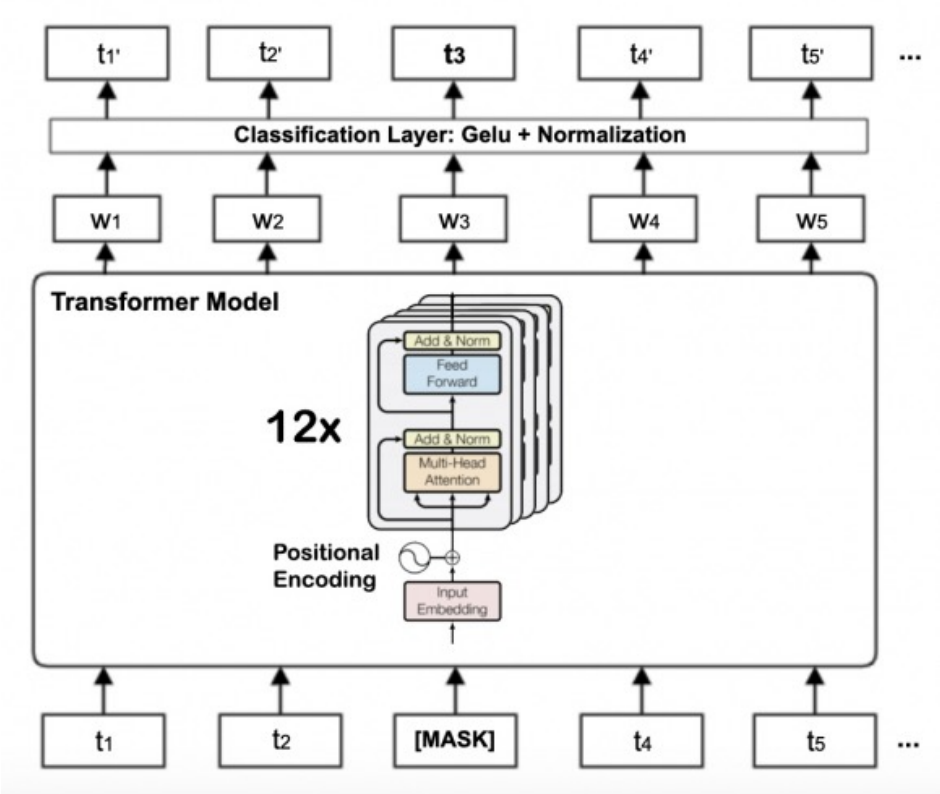
BERT and some of its variants

BERT

BERT

Bidirectional Encoder
Representation of Transformers

- Transformer architecture
- Encoders-only
- Pretrained architecture



Pretrained model

PRE-TRAINING

- On large public datasets
- Specific *pretext* tasks:
  - MLM: Masked language modeling
  - NSP: Next sentence prediction
- Slow
- Random initialization

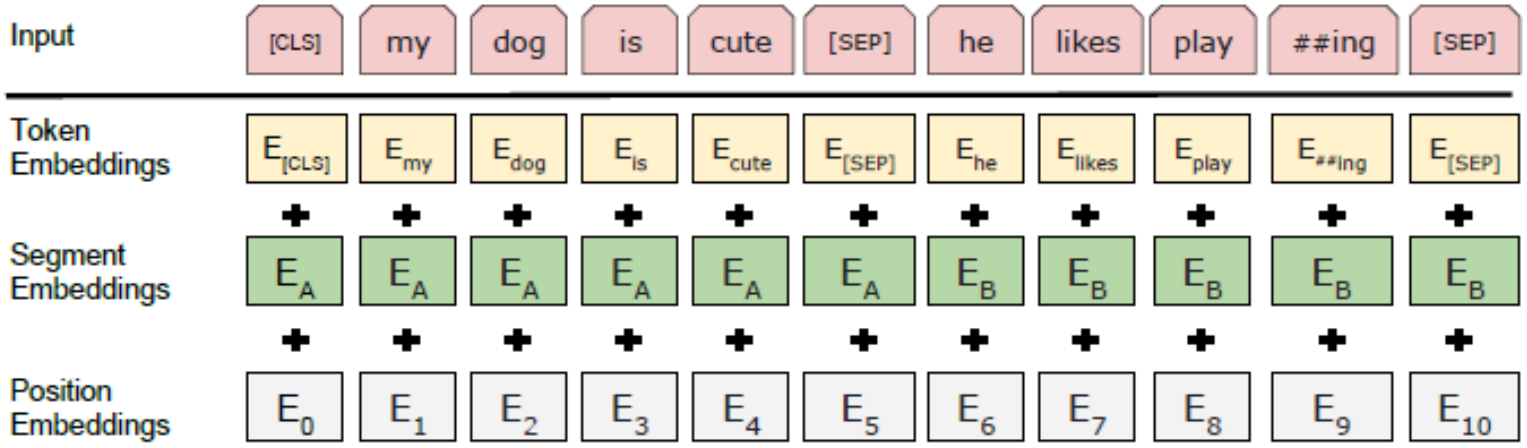
FINE-TUNING

- On task-specific datasets
- Downstream tasks
- Much faster
- Pre-training as initialization

Inputs

- Input sequence: 1 or 2 sentences
  - Sequence starts with [CLS]
  - Sentence ends with [SEP]
- combined length is ≤ 512 tokens

- Embedding layer –fully learnable
  - Token embedding
  - Sentence embedding
  - Positional embedding



MLM task

Given a masked token, predict the original input token

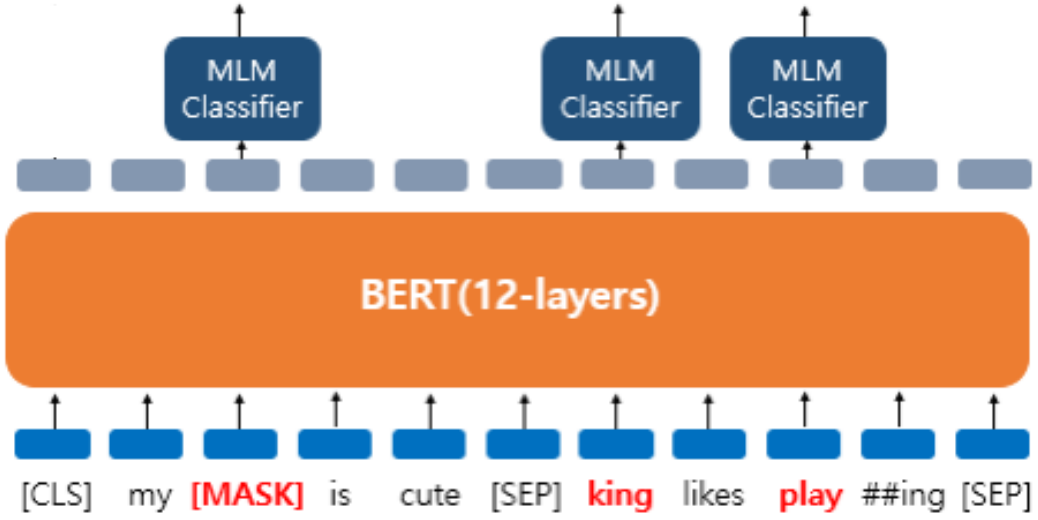
15% of input tokens are replaced with an auxiliary [mask] token of which:

- 80% masked
- 10% replaced with another token
- 10% not replaced

Static masking: only done at the beginning

MLM task

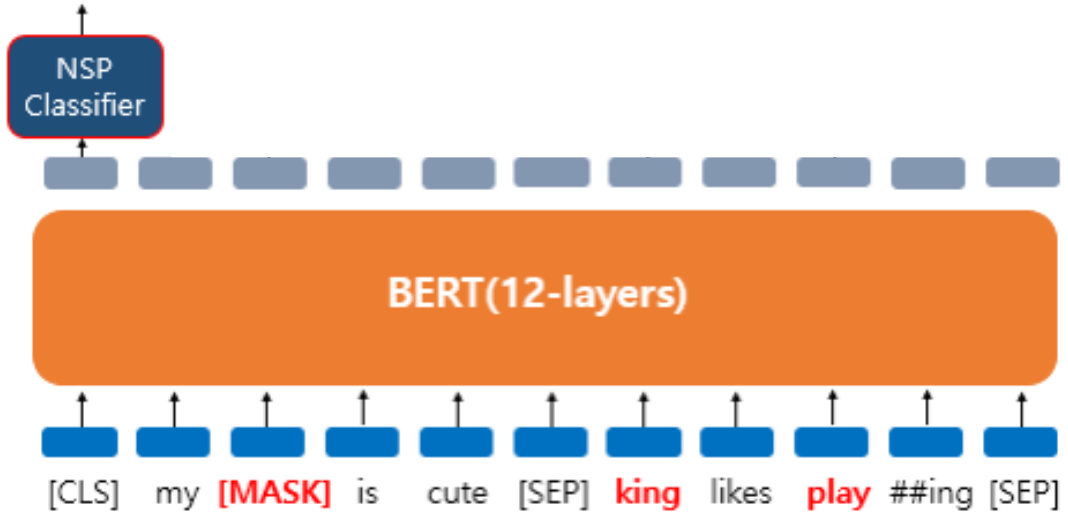
MLM is responsible for bidirectionality!



NSP task

Given two sentences A and B, predict if B IsNext or NotNext to A

NSP provides understanding of relationships between sentences



Parameters

$BERT_{BASE}$ 12 layers, Hidden size=768, 12 Attention heads, 110M Total Parameters

$BERT_{LARGE}$ 24 layers, Hidden size =1024, 16 Attention heads, 340M Total Parameters

RoBERTa

Robustly Optimized BERT Approach

RoBERTa vs BERT

Same architecture as $\text{BERT}_{\text{LARGE}}$

with some design differences

- Dynamic masking
- No NSP task
- Larger training batches
- 50k BPE vocabulary

Dynamic masking vs Static masking

Static masking: all data is masked at once

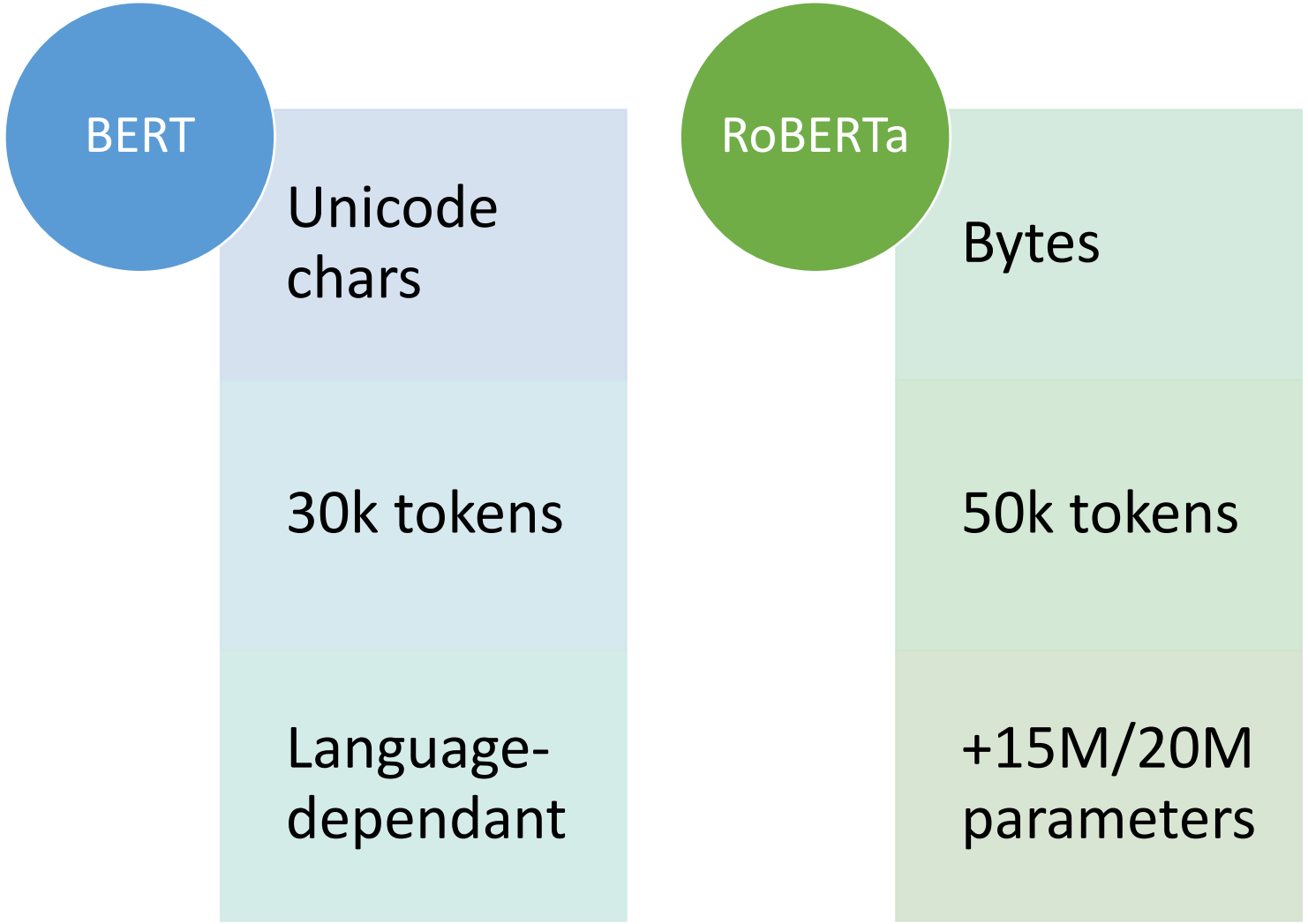
Dynamic masking: masking is generated when a sequence is passed to the model

Larger Batches

Large batch training can **improve optimization speed** and end-task performance **if the learning rate is increased** appropriately

bsz	steps	lr	ppl	MNLI-m	SST-2
256	1M	1e-4	3.99	84.7	92.7
2K	125K	7e-4	**3.68**	**85.2**	**92.9**
8K	31K	1e-3	3.77	84.6	92.8

BPE (Byte-Pair Encoding) vocabulary



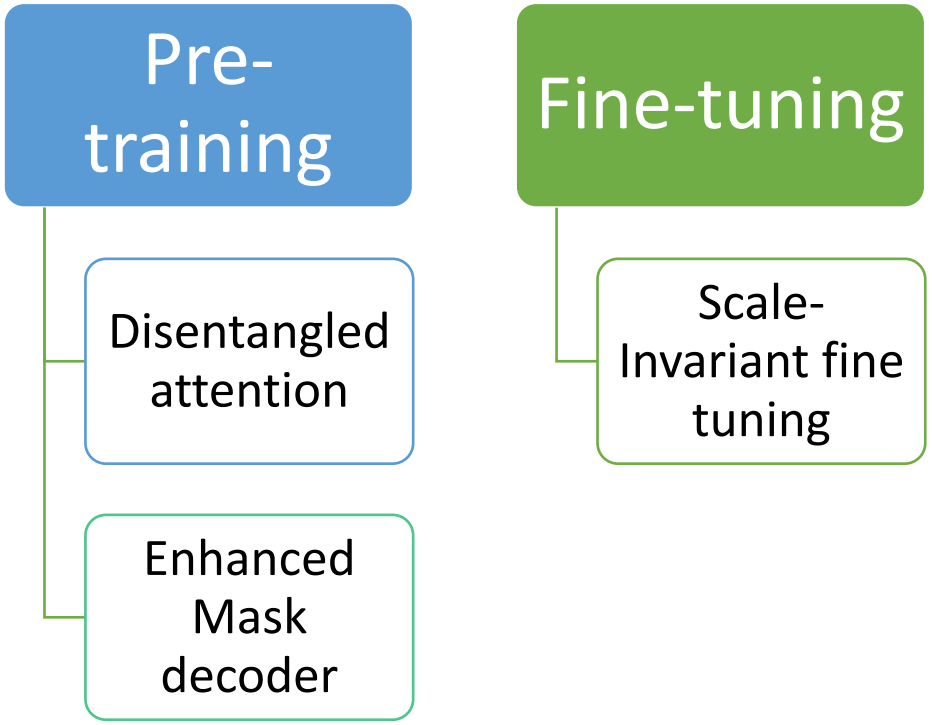
DeBERTa

Decoder Enhanced BERT with disentangled Attention

DeBERTa vs (Ro)BERT(a)

Mostly built on RoBERTa

design differences



Disentangled self-Attention

token in position *i* in the sentence is represented using two vectors:

- $\{H_i\}$ content of token
- $\{P_{i|J}\}$ position of token w.r.t token at position *j*

Attention score can be computed as $\mathrm{A_{ij}} = \{\mathrm{H_i}, \mathrm{P_{i|j}}\} \times \{\mathrm{H_j}, \mathrm{P_{j|i}}\}^{\mathrm{T}}$

$$= H_i H_j^T + H_i P_{j|i}^T + P_{i|j} H_j + P_{i|j} P_{j|i}^T$$

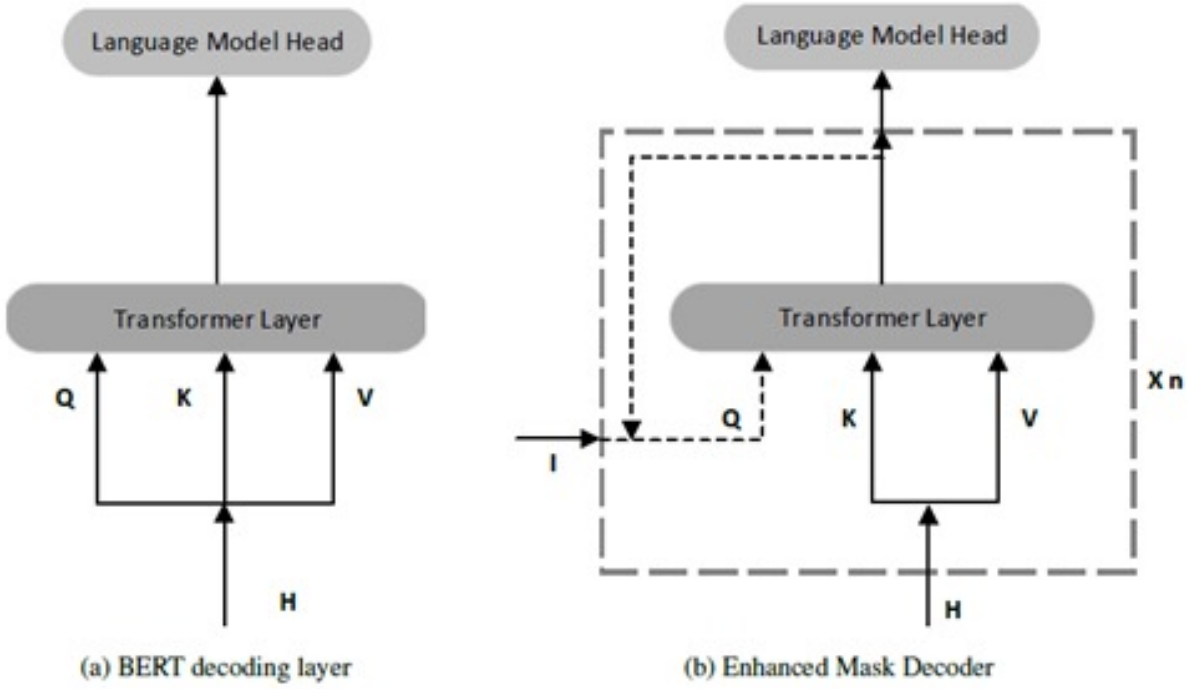
Disentangled self-Attention

$$A_{i,j} = \underbrace{H_i H_j^T}_{content\text{-}to\text{-}content} + \underbrace{H_i P_{j|i}^T}_{content\text{-}to\text{-}position} + \underbrace{P_{i|j}\, H_j}_{position\text{-}to\text{-}content} + \underbrace{P_{i|j} P_{j|i}^T}_{position\text{-}to\text{-}position}$$

More specifically

$$\tilde{A}_{i,j} = \underbrace{Q_i^c K_j^{cT}}_{content\text{-}to\text{-}content} + \underbrace{Q_i^c K_{\delta(i,j)}^{rT}}_{content\text{-}to\text{-}position} + \underbrace{K_j^c Q_{\delta(j,i)}^{rT}}_{position\text{-}to\text{-}content}$$

Enhanced Mask decoder



Absolute position of a token in a sentence NOT captured in disentangled attention

EMD Layer Enhanced Mask decorder layer

- Between transformer layer and softmax output layer
- Captures absolute positions

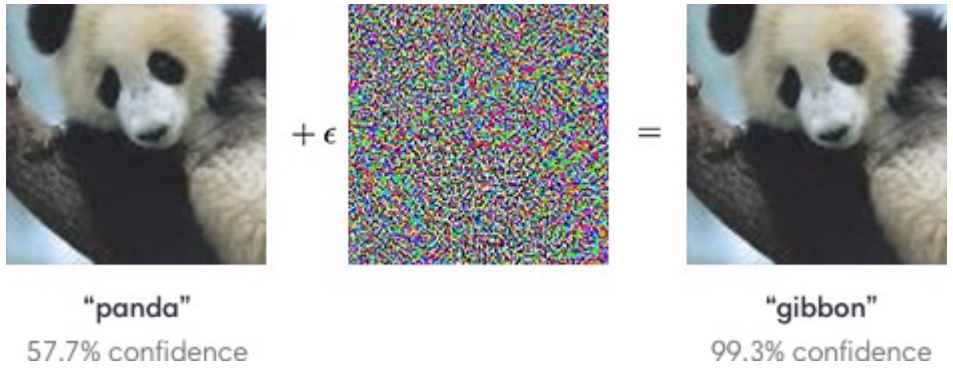
SiFT

Scale-invariant Fine-Tuning

A form of adversarial training

- Improves robustness to adversarial examples
- Is a form of regularization

PROBLEM: Instability of adversarial training grows with model size



SiFT applies the perturbation to normalized embedding vectors

DistilBERT

Knowledge distillation

A larger **teacher** model trains a smaller **student** model

They operate on the same dataset

How?

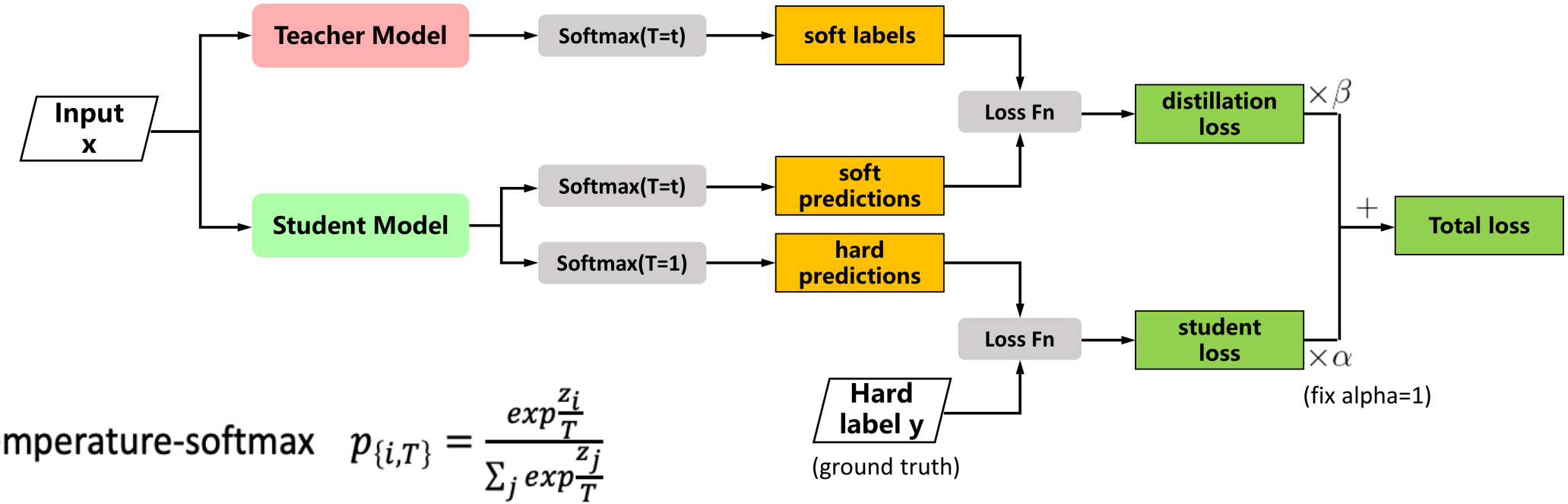
Knowledge distillation

A larger **teacher** model trains a smaller **student** model

They operate on the same dataset

How? Through the loss function!

Knowledge distillation



Temperature-softmax $p_{\{i,T\}} = \frac{exp\frac{z_i}{T}}{\sum_j exp\frac{z_j}{T}}$

Distillation Loss: $L_d = \sum_i t_{i,T} log(s_{i,T})$

Student Loss: $L_s = \sum_i y_i log(s_i)$
T=1

Knowledge distillation for DistilBERT

Training loss = Distillation loss + Student loss

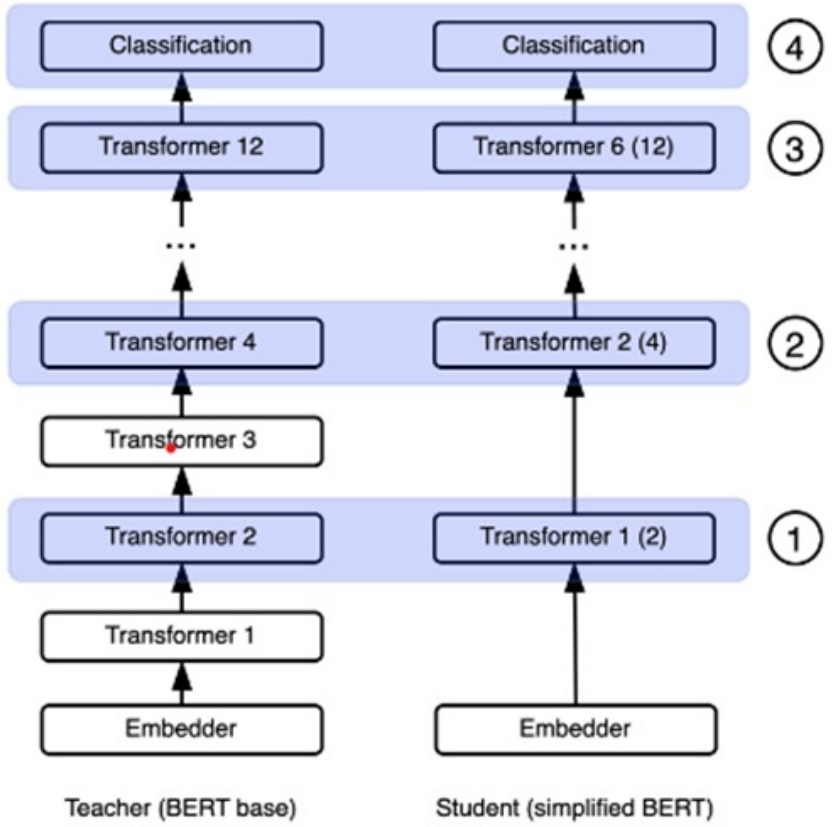
$$L = \alpha L_d + \beta L_s$$

DistilBERT Training loss + Cosine Embedding loss

Architecture

The initialized values of parameters come from the teacher model

½ layers w.r.t BERT !



Performance

DistilBERT has 40% fewer parameters than $BERT_{BASE}$ (66M vs 100M) and is 60% faster. It also outperforms ELMo and retains 97% of the capabilities of BERT.

Model	**Score**
ELMo	68.7
BERT-base	79.5
DistilBERT	77.0

On GLUE benchmark

Model	IMDb (acc.)	SQuAD (EM/F1)
BERT-base	93.46	81.2/88.5
DistilBERT	92.82	77.7/85.8
DistilBERT (D)	-	79.1/86.9

On some downstream tasks

Model	# param. (Millions)	Inf. time (seconds)
ELMo	180	895
BERT-base	110	668
DistilBERT	66	410

While being more efficient

Implementation for spoiler detection

Goodreads Dataset

- Reviews: 1,378,033 (#rows)
- Books: 25,475
  Users: 18,892

	user_id	timestamp	review_sentences	rating	has_spoiler	book_id	review_id
0	8842281e1d1347389f2ab93d60773d4d	2017-08-30	[[0, This is a special book.], [0, It started ...	5	True	18245960	dfdbb7b0eb5a7e4c26d59a937e2e5feb
1	8842281e1d1347389f2ab93d60773d4d	2017-03-22	[[0, Recommended by Don Katz.], [0, Avail for ...	3	False	16981	a5d2c3628987712d0e05c4f90798eb67
2	8842281e1d1347389f2ab93d60773d4d	2017-03-20	[[0, A fun, fast paced science fiction thrille...	3	True	28684704	2ede853b14dc4583f96cf5d120af636f
3	8842281e1d1347389f2ab93d60773d4d	2016-11-09	[[0, Recommended reading to understand what is...	0	False	27161156	ced5675e55cd9d38a524743f5c40996e
4	8842281e1d1347389f2ab93d60773d4d	2016-04-25	[[0, I really enjoyed this book, and there is ...	4	True	25884323	332732725863131279a8e345b63ac33e
...	...	...	...	...	...	...	...
1378028	35cef391b171b4fca45771e508028212	2013-04-16	[[0, Can't wait for Travis' POV], [0, Travis B...	0	False	15745950	0e1db3d4b04256f9660f5d276ddf1314
1378029	35cef391b171b4fca45771e508028212	2012-12-28	[[0, Had this on my to-read shelf forever.], [...	0	False	10861195	0b7f352e58caf0fd1f961e98ef04e89c
1378030	35cef391b171b4fca45771e508028212	2013-03-25	[[0, The last book left me wanting for more.],...	4	False	6131164	9b19eff33ddb14e9e68fca2e90379e46
1378031	35cef391b171b4fca45771e508028212	2013-01-24	[[0, Things are heating up in the second novel...	4	False	10025305	8be463fed78f0da63e964706f710332b
1378032	35cef391b171b4fca45771e508028212	2012-12-29	[[0, Before I even start this review, I must s...	5	True	6482837	62ed1263c7d216986cc419cd4e8a408b

1378033 rows × 7 columns

Preprocessing

- Reduced-size dataset --> 20% of original data
- Balancing the dataset
- Flattening -- > all reviews of a user as a single review of that user
- Lowercase letters
- Removing any symbol, punctuation, white spaces, stop words and digits (e.g. ? ! # * () [])

```
size of dataset: 275607
nonspoiler: 257603
spoiler: 18004
weight-decreased nonspoiler: 20608
balanced total: 38612
```

```
'[0, \'3 1/2 stars.\'][0, "Like most people here I agree it\'s a futuristic pretty little liars/gossip girl book."][0, "We\'ve got the perfect girl with her secret."][1, \'(Which is kinda disturbing She likes her "brother" ] ) A girl who\\\'s loony and has a drug problem.\'][0, "A girl who\'s life crumbles and she has to move lower (essentially means she\'s poor) A poor girl with a drug dealing boyfriend and a boy with a supercomputer inside his mind."][0, "Overall the book was pretty enjoyable though I wished it had more world building since I couldn\'t understand that aspect of the story like why the tower was built."][0, \'I will read the next book when it comes out though!\']'
```

```
'stars like people agree futuristic pretty little liars gossip girl book weve got perfect girl secret kinda disturbing likes brother girl whos loony drug problem girl whos life crumbles move lower essentially means shes poor poor girl drug dealing boyfriend boy supercomputer inside mind overall book pretty enjoyable though wished world building since couldnt understand aspect story like tower built read next book comes though'
```

Preprocessing

Summary of the dataset after preprocessing

	class	review_length
count	38612.000000	38612.000000
mean	0.466280	134.084637
std	0.498868	138.094813
min	0.000000	0.000000
25%	0.000000	36.000000
50%	0.000000	91.000000
75%	1.000000	188.000000
max	1.000000	1427.000000

95% percentile of the length of reviews = 401

We drop reviews with less than 20 and more than 500 length

Train, Test, and Validation datasets

- Train 70% -- > 22426
- Test 20% --> 1922
- Validation 10% --> 769

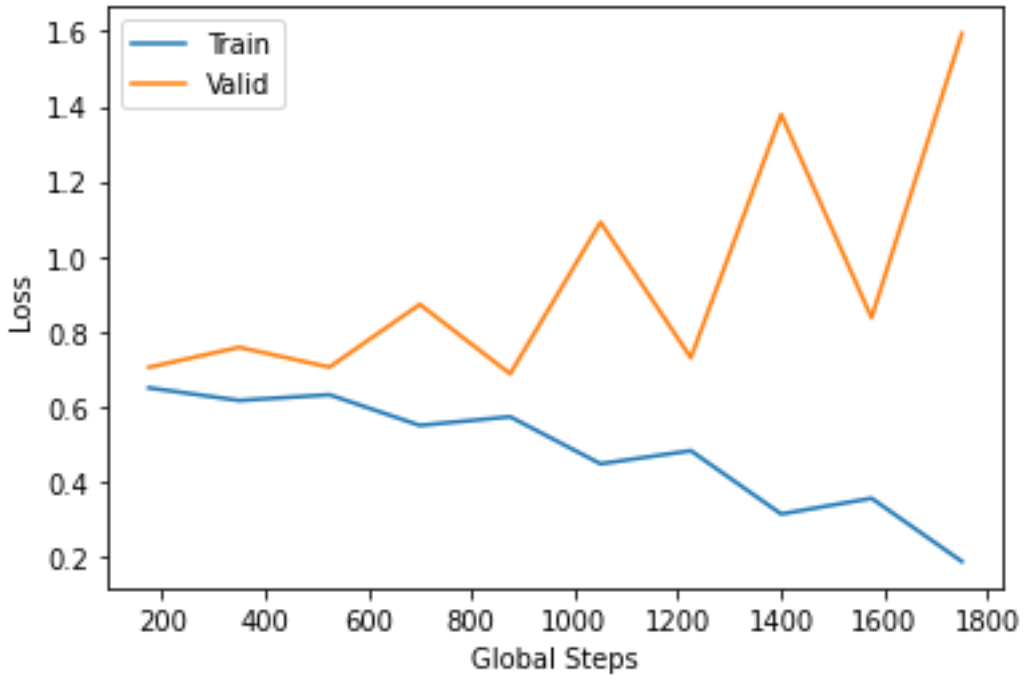
95% percentile of the length of reviews in training dataset= 363

LSTM Model

- Device : CPU
- Tokenizer : Spacy
- Batch_size: 64
- Num_epochs : 5
- Optimizer : ADAM
- Learning_Rate: 0.001
- Input_size : 300
- Hidden_size: 128
- Embedding_dimension: 300
- Bi-directional LSTM
- Dropout layer P= 0.5
- Binary Cross Entropy Loss

```
Epoch [1/5], Step [175/1755], Train Loss: 0.6493, Valid Loss: 0.7039
Model saved to ==> /content/drive/My Drive/DL/DL_Final_Project/destination_lstm/model.pt
Model saved to ==> /content/drive/My Drive/DL/DL_Final_Project/destination_lstm/metrics.pt
Epoch [1/5], Step [350/1755], Train Loss: 0.6156, Valid Loss: 0.7574
Epoch [2/5], Step [525/1755], Train Loss: 0.6309, Valid Loss: 0.7045
Epoch [2/5], Step [700/1755], Train Loss: 0.5491, Valid Loss: 0.8719
Epoch [3/5], Step [875/1755], Train Loss: 0.5721, Valid Loss: 0.6867
Model saved to ==> /content/drive/My Drive/DL/DL_Final_Project/destination_lstm/model.pt
Model saved to ==> /content/drive/My Drive/DL/DL_Final_Project/destination_lstm/metrics.pt
Epoch [3/5], Step [1050/1755], Train Loss: 0.4467, Valid Loss: 1.0911
Epoch [4/5], Step [1225/1755], Train Loss: 0.4816, Valid Loss: 0.7294
Epoch [4/5], Step [1400/1755], Train Loss: 0.3128, Valid Loss: 1.3784
Epoch [5/5], Step [1575/1755], Train Loss: 0.3548, Valid Loss: 0.8363
Epoch [5/5], Step [1750/1755], Train Loss: 0.1861, Valid Loss: 1.5946
Model saved to ==> /content/drive/My Drive/DL/DL_Final_Project/destination_lstm/metrics.pt
Finished Training!
CPU times: user 2h 29min 6s, sys: 14min 54s, total: 2h 44min 1s
Wall time: 2h 43min 18s
```

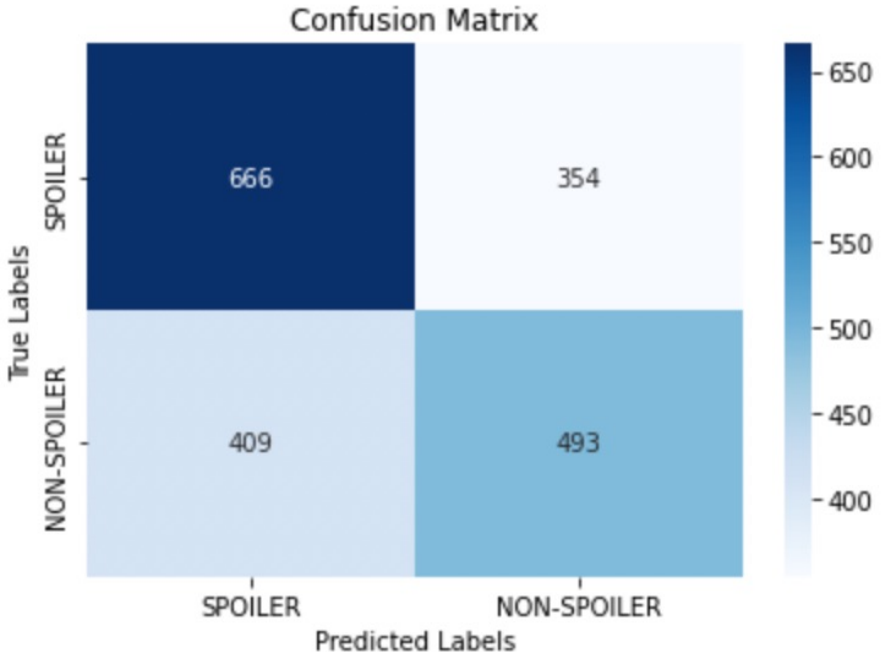
LSTM Model



Classification Report:

	precision	recall	f1-score	support
1	0.6195	0.6529	0.6358	1020
0	0.5821	0.5466	0.5638	902
accuracy			0.6030	1922
macro avg	0.6008	0.5998	0.5998	1922
weighted avg	0.6019	0.6030	0.6020	1922

AUC= 0.5997521846876223

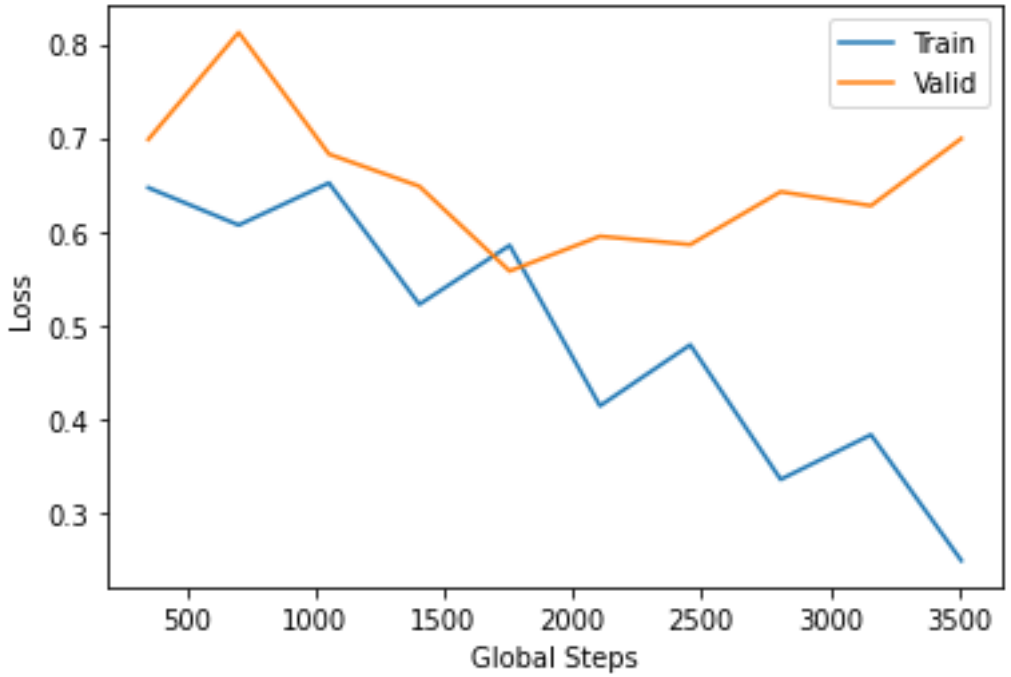


BERT Model

- Device : CUDA
- Tokenizer : BERT Tokenizer
- Max_Seq_Len : 315
- Batch_size: 32
- Num_epochs : 5
- Optimizer : ADAM
- Learning_Rate: 2e-5
- Binary Cross Entropy Loss

```
Epoch [1/5], Step [351/3510], Train Loss: 0.6465, Valid Loss: 0.6981
Model saved to ==> /content/drive/My Drive/DL/destination/model.pt
Model saved to ==> /content/drive/My Drive/DL/destination/metrics.pt
Epoch [1/5], Step [702/3510], Train Loss: 0.6067, Valid Loss: 0.8124
Epoch [2/5], Step [1053/3510], Train Loss: 0.6516, Valid Loss: 0.6824
Model saved to ==> /content/drive/My Drive/DL/destination/model.pt
Model saved to ==> /content/drive/My Drive/DL/destination/metrics.pt
Epoch [2/5], Step [1404/3510], Train Loss: 0.5220, Valid Loss: 0.6481
Model saved to ==> /content/drive/My Drive/DL/destination/model.pt
Model saved to ==> /content/drive/My Drive/DL/destination/metrics.pt
Epoch [3/5], Step [1755/3510], Train Loss: 0.5850, Valid Loss: 0.5576
Model saved to ==> /content/drive/My Drive/DL/destination/model.pt
Model saved to ==> /content/drive/My Drive/DL/destination/metrics.pt
Epoch [3/5], Step [2106/3510], Train Loss: 0.4137, Valid Loss: 0.5947
Epoch [4/5], Step [2457/3510], Train Loss: 0.4788, Valid Loss: 0.5859
Epoch [4/5], Step [2808/3510], Train Loss: 0.3351, Valid Loss: 0.6423
Epoch [5/5], Step [3159/3510], Train Loss: 0.3828, Valid Loss: 0.6273
Epoch [5/5], Step [3510/3510], Train Loss: 0.2485, Valid Loss: 0.6990
Model saved to ==> /content/drive/My Drive/DL/destination/metrics.pt
Finished Training!
CPU times: user 1h 38min 31s, sys: 24min 46s, total: 2h 3min 18s
Wall time: 2h 3min 24s
```

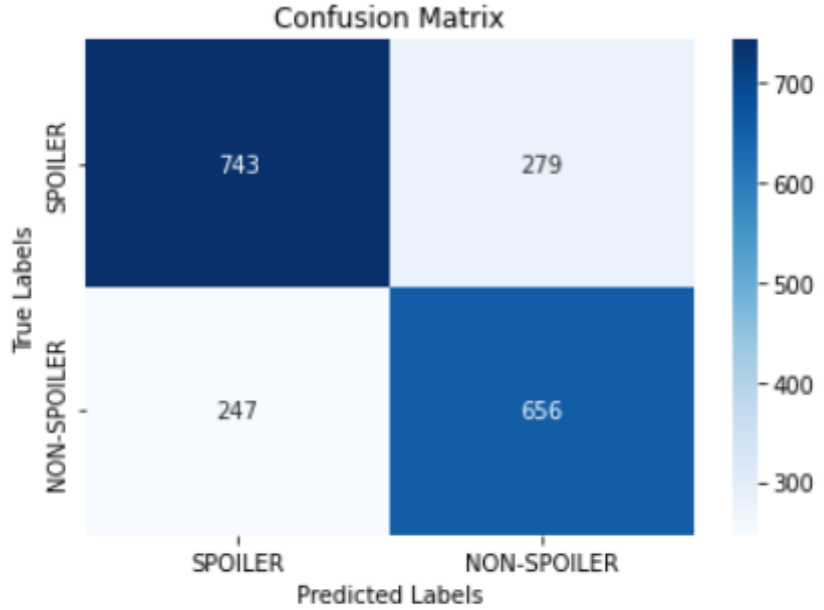
BERT Model



Classification Report:

	precision	recall	f1-score	support
1	0.7505	0.7270	0.7386	1022
0	0.7016	0.7265	0.7138	903
accuracy			0.7268	1925
macro avg	0.7261	0.7267	0.7262	1925
weighted avg	0.7276	0.7268	0.7270	1925

AUC= 0.7267366009799906



BIBLIOGRAPHY

Karpathy, Andrej. “The Unreasonable Effectiveness of Recurrent Neural Networks”. 2015. karpathy.github.io/2015/05/21/rnn-effectiveness/. Accessed 24 June 2022. http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Zhu, Yukun, et al. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. 2015. doi:10 . 48550 /ARXIV.1506.06724. https://arxiv.org/abs/1506.06724

Vaswani, Ashish, et al. Attention Is All You Need. 2017. doi:10.48550/ARXIV.1706.03762. https://arxiv.org/abs/1706.03762

Devlin, Jacob, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018. doi:10.48550/ARXIV.1810.04805. https://arxiv.org/abs/1810.04805

Ilić, Suzana, et al. Deep contextualized word representations for detecting sarcasm and irony. 2018. doi:10.48550/ARXIV.1809.09795. https://arxiv.org/abs/1809.09795

Ott, Myle, et al. Scaling Neural Machine Translation. 2018. doi:10 . 48550 / ARXIV.1806.00187. https://arxiv.org/abs/1806.00187

Aguilar, Gustavo, et al. Knowledge Distillation from Internal Representations. 2019. doi:10.48550/ARXIV.1910.03723. https://arxiv.org/abs/1910.03723

BIBLIOGRAPHY

Liu, Yinhan, et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019. doi:10.48550/ARXIV.1907.11692. https://arxiv.org/abs/1907.11692

Sanh, Victor, et al. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. 2019. doi:10.48550/ARXIV.1910.01108. https://arxiv.org/pdf/1910.01108.pdf

Wan, Mengting, et al. Fine-Grained Spoiler Detection from Large-Scale Review Corpora. 2019. doi:10.48550/ARXIV.1905.13416.16. https://arxiv.org/abs/1905.13416

Hatzel, Hans Ole. “Using Neural Language Models to Detect Spoilers”. 2020. Universitat Hamburg, MA thesis.
He, Pengcheng, et al. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. 2020. doi:10.48550/ARXIV.2006.03654. https://www.inf.uni-hamburg.de/en/inst/ab/lt/teaching/theses/completed-theses/2020-ma-hatzel-spoiler.pdf

Minaee, Shervin, et al. Deep Learning Based Text Classification: A Comprehensive Review. 2020. doi:10.48550/ARXIV.2004.03705. https://arxiv.org/abs/2004.03705

Singh, Harsh and Qusay Mahmoud. “NLP-Based Approach for Predicting HMI State Sequences Towards Monitoring Operator Situational Awareness”. Sensors, vol. 20, June 2020, p. 3228. doi:10.3390/s20113228. https://www.mdpi.com/1424-8220/20/11/3228

BIBLIOGRAPHY

Xia, Patrick, et al. “Which *BERT? A Survey Organizing Contextualized Encoders”. “Which *BERT? A Survey Organizing Contextualized Encoders”. Jan. 2020, pp. 7516–7533. https://arxiv.org/abs/2010.00854

Zhang, Aston, et al. Dive into Deep Learning. https://d2l.ai, 2020. https://d2l.ai/

Cahuantzi, Roberto, et al. A comparison of LSTM and GRU networks for learning symbolic sequences. 2021. doi:10.48550/ARXIV.2107.02248. https://arxiv.org/abs/2107.02248

Khalid, Usama, et al. RUBERT: A Bilingual Roman Urdu BERT Using Cross Lingual Transfer Learning. 2021. doi:10.48550/ARXIV.2102.11278. https://arxiv.org/abs/2102.11278

Singh, Aastha. “Evolving with BERT: Introduction to RoBERTa”. 2021. medium.com/analytics-vidhya/evolving-with-bert-introduction-to-roberta-5174ec0e7c82. Accessed 20 June 2022. https://medium.com/analytics-vidhya/evolving-with-bert-introduction-to-roberta-5174ec0e7c82

Wang, Benyou, et al. “On Position Embeddings in {BERT}”. International Conference on Learning Representations, 2021. https://openreview.net/forum?id=onxoVA9FxMw

BIBLIOGRAPHY

Singh, Arjun, et al. “Evolving Long Short-Term Memory Network-Based Text Classification ”. Computational Intelligence and Neuroscience, vol. 2022, 2022, p. 11. doi:https://doi.org/10.1155/2022/472563 https://www.hindawi.com/journals/cin/2022/4725639/

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014. https://nlp.stanford.edu/pubs/glove.pdf

Mikolov, Tomas, et al. "Advances in pre-training distributed word representations." arXiv preprint arXiv:1712.09405 (2017). https://arxiv.org/abs/1712.09405