Cross-Domain Acronym Disambiguation

Gaurav Agarwal (BT17CSE001)
Abhibha Gupta (BT17CSE020)
Himabindu Thogaru (BT17CSE056)

Submitted to
Dr. Tausif Diwan
Assistant Professor (CSE)

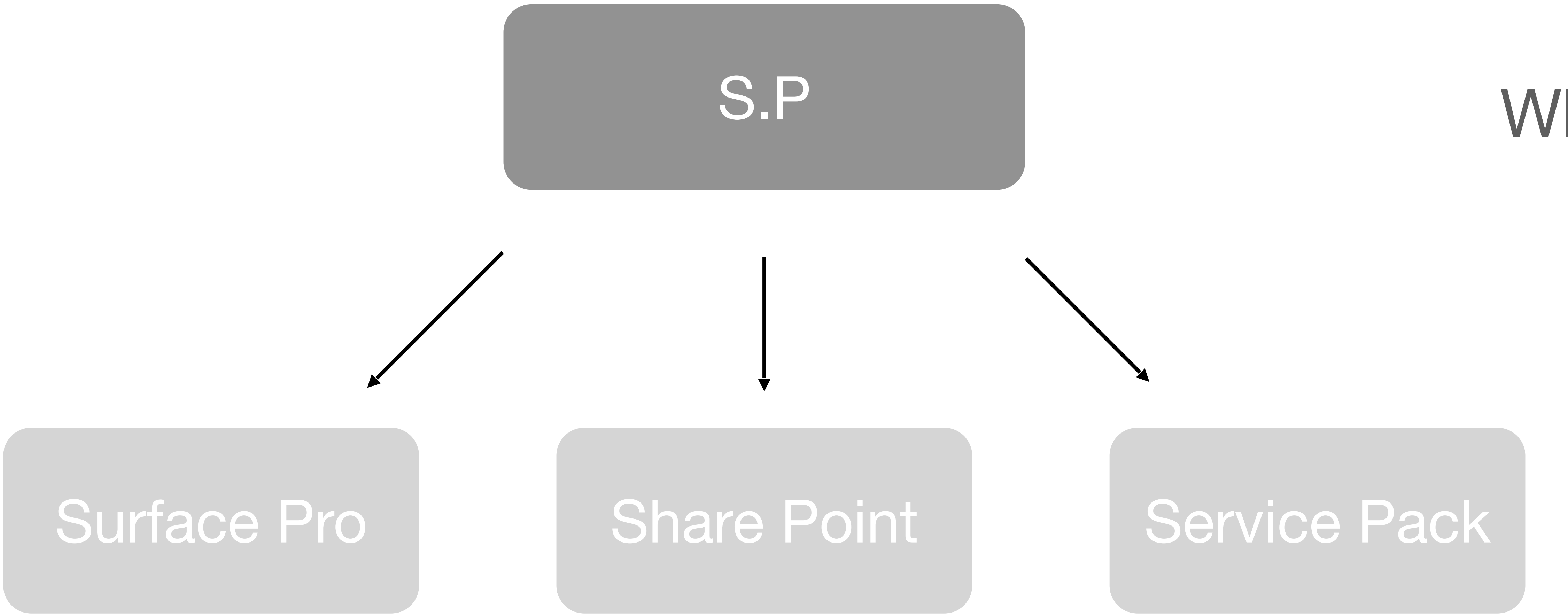
Index

Problem Statement

Acronym Disambiguation

Identification of ambiguous acronyms is a major challenge in information retrieval and analysis.

According to Liu et al, almost 81% acronyms used in MEDLINE are ambiguous.



Which definition to choose?

Ambiguous Definitions

Applications

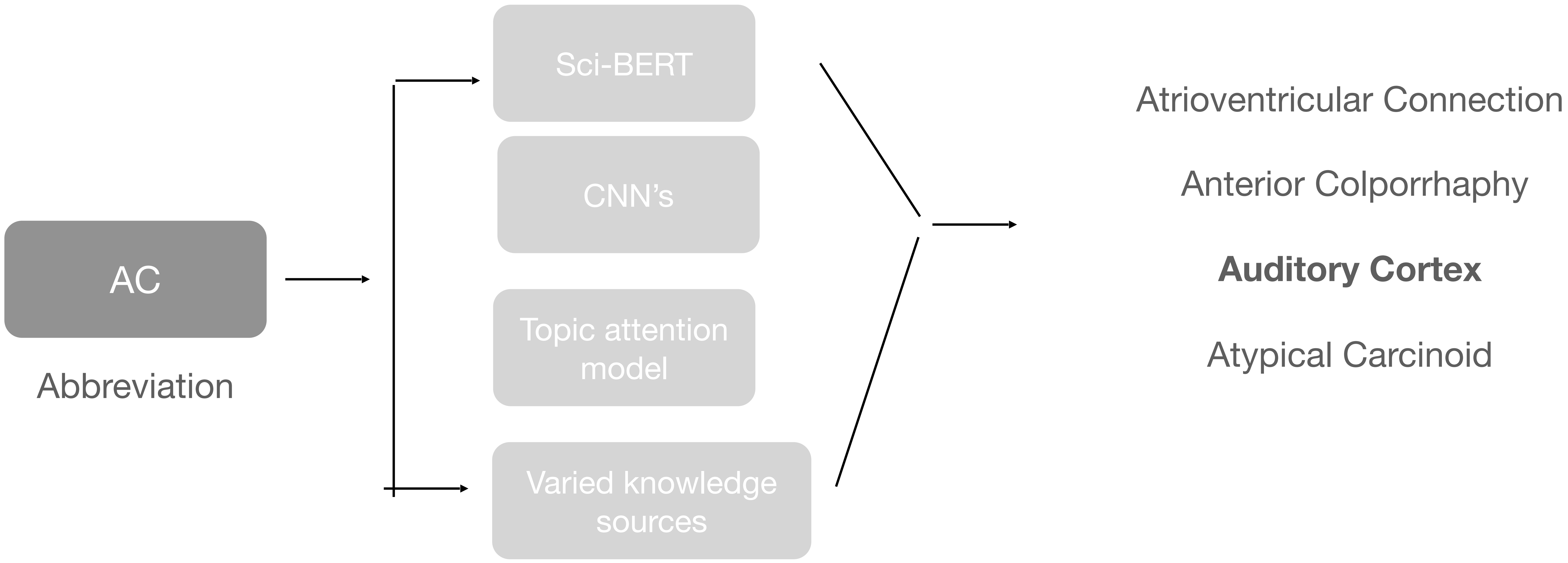
- Information Retrieval
- Machine Translation
- Text understanding
- Text summarization

Previous Work

Medical/Biological Term Disambiguation

- Most of the existing work centers around disambiguating abbreviations in biological and medical texts. (Liu et al ; Joopudi et al)

- Researchers have used Sci-BERT, a variant of BERT pre-trained on medical texts for acronym disambiguation.

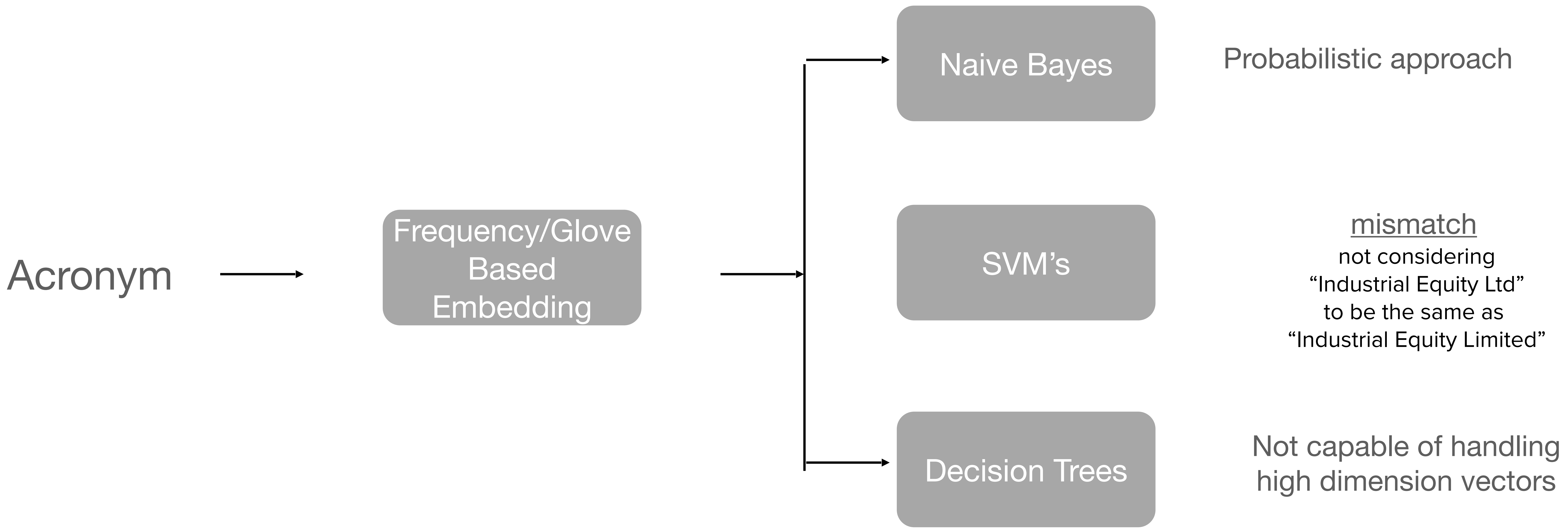


Previous Work

Gardner et al. (2017) -Base Paper

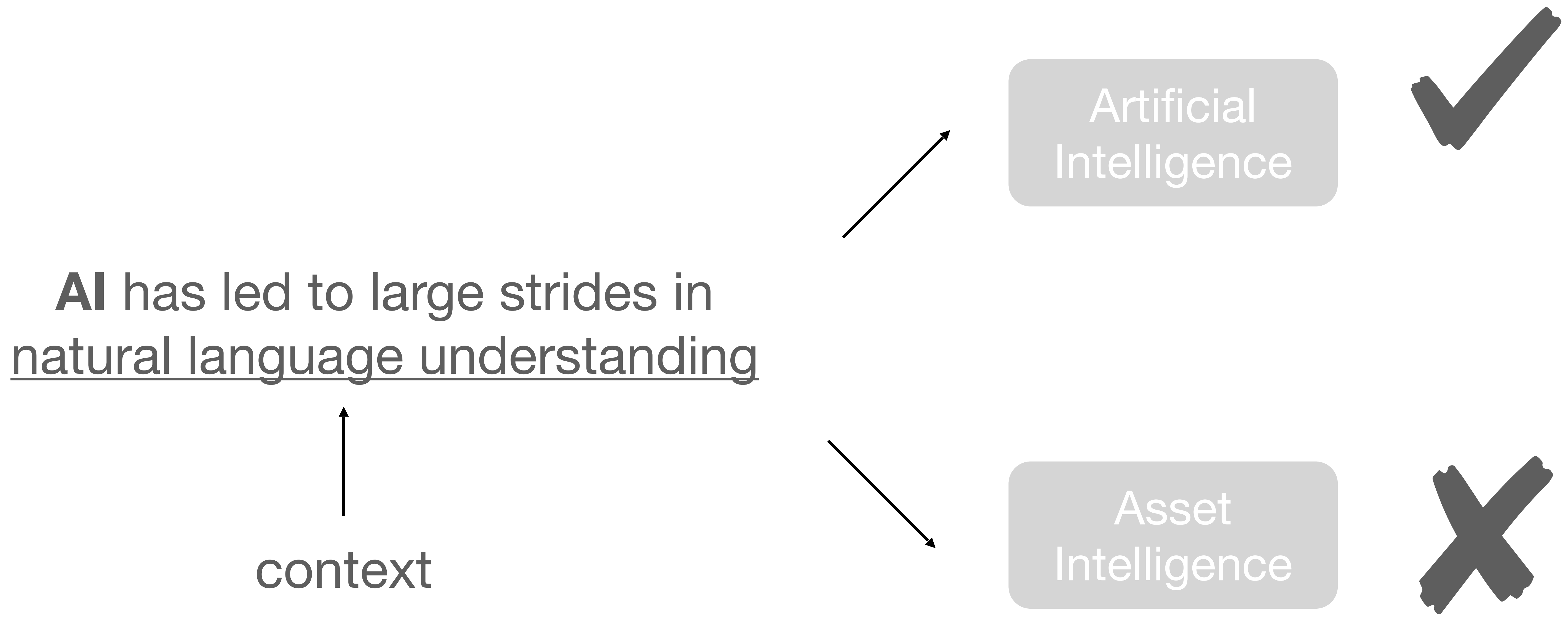
- The researchers experiment with primitive machine learning classifiers like Naive Bayes, Support Vector Machines, Decision trees.

- Their own dataset derived from Wikipedia(by analysing webpage source code).



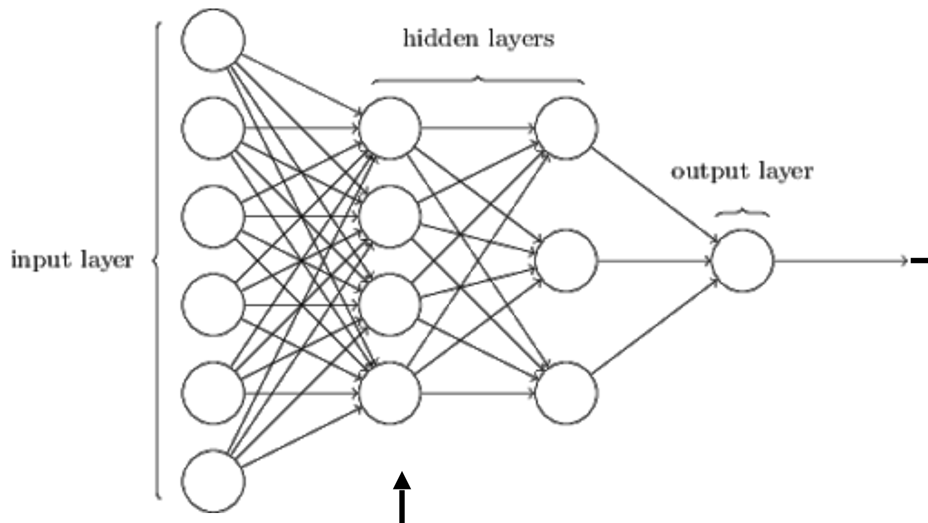
Our Approach

- Design a machine-learning classifier to match ambiguous acronyms with accurate definitions based on **context**.
- Develop an end to end pipeline that is capable of disambiguating cross domain abbreviations.



Our Approach

United States of America



- Baselines: ML classifiers (NB, SVM)
- Transfer Learning (BERT, UlmFit)

Create Embeddings

- Pre-trained (GloVe, FastText)
- Frequency based (Tf-Idf)
- Custom Trained Embeddings

I live in US

Words in a window

My brother and I live in US

Dataset

AAAI-21 - Shared Task

- Our dataset was published as part of the AAAI-21 shared task on Acronym Disambiguation.
- The dataset consists of 4 files
  - train.json
  - test.json
  - dev.json
  - diction.json

Dataset

{'acronym': 4,

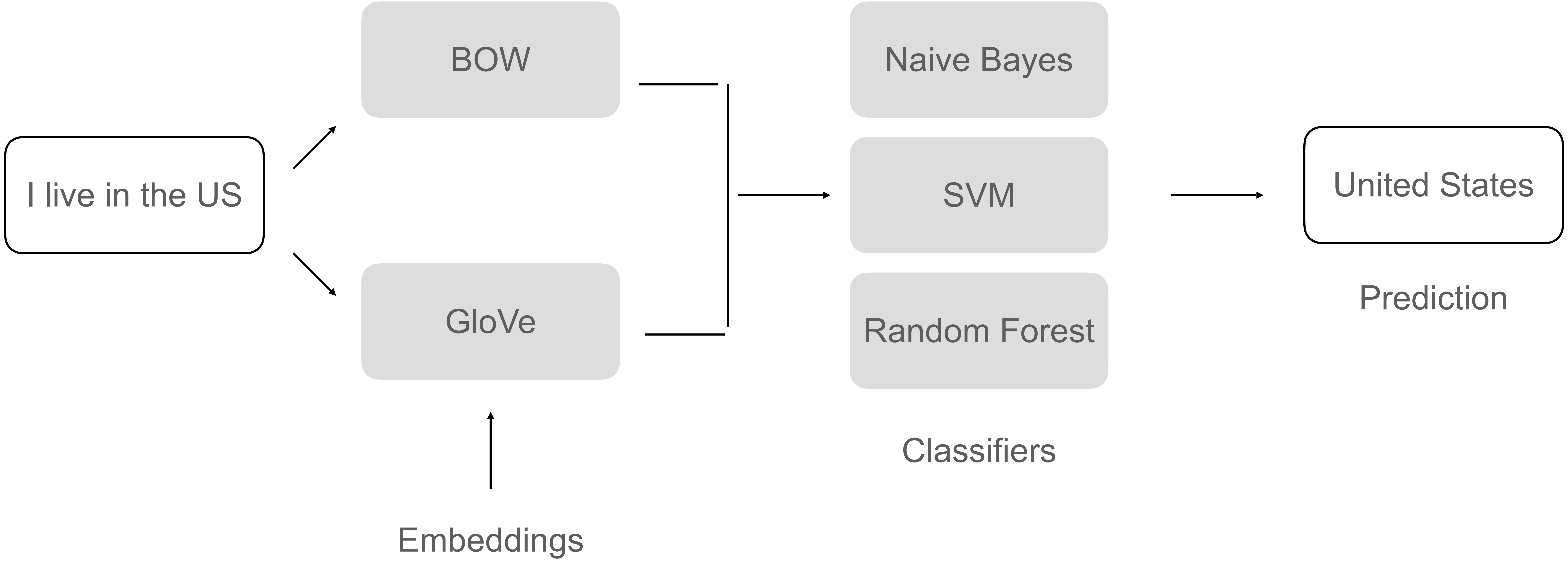
'expansion': ‘United States of America',

'id': 'TR-0',

'tokens': [‘I’, ‘live’, ‘in’, ‘the’, ‘USA’, ‘.’]}

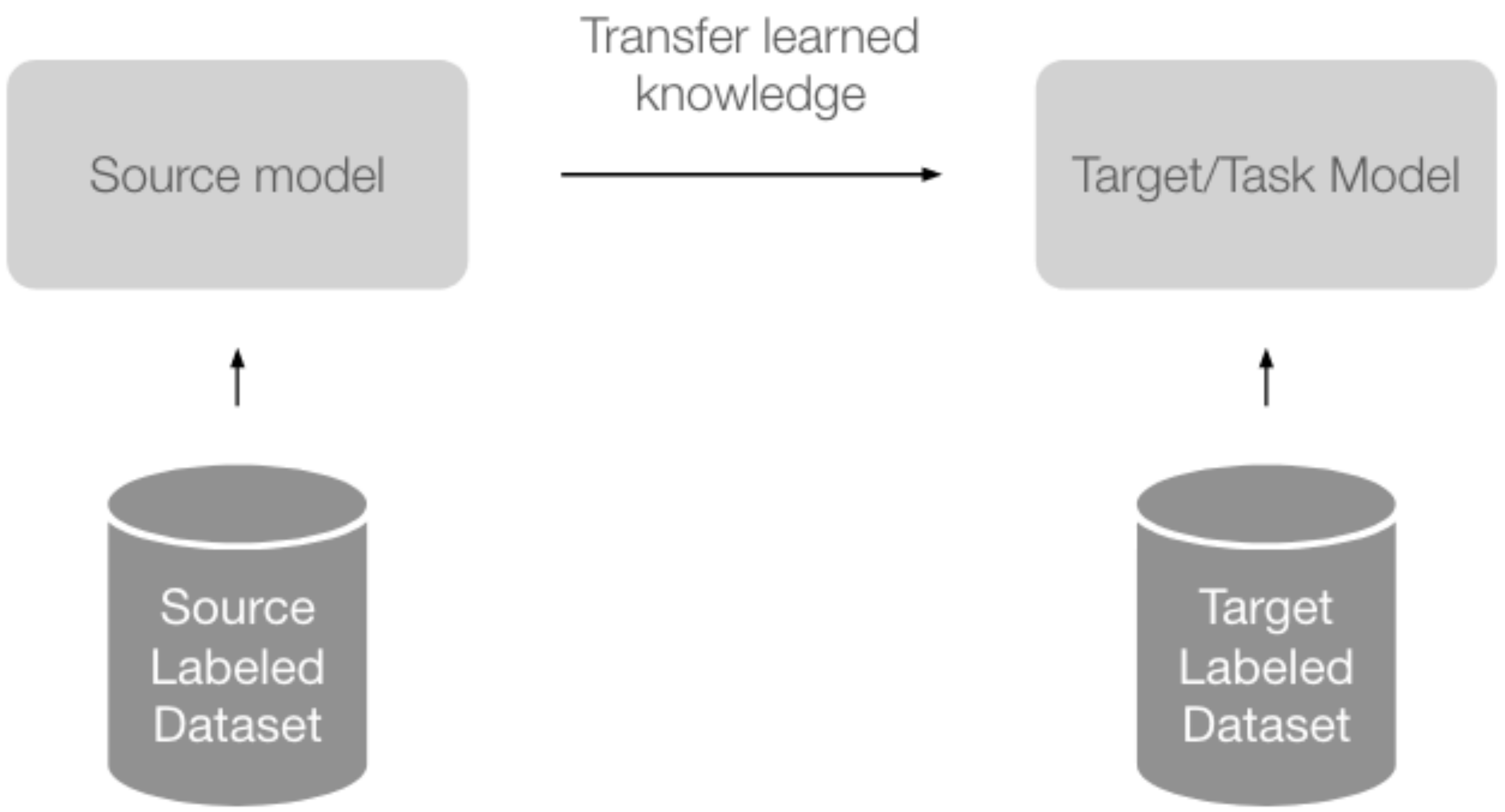
Methodology

Baselines

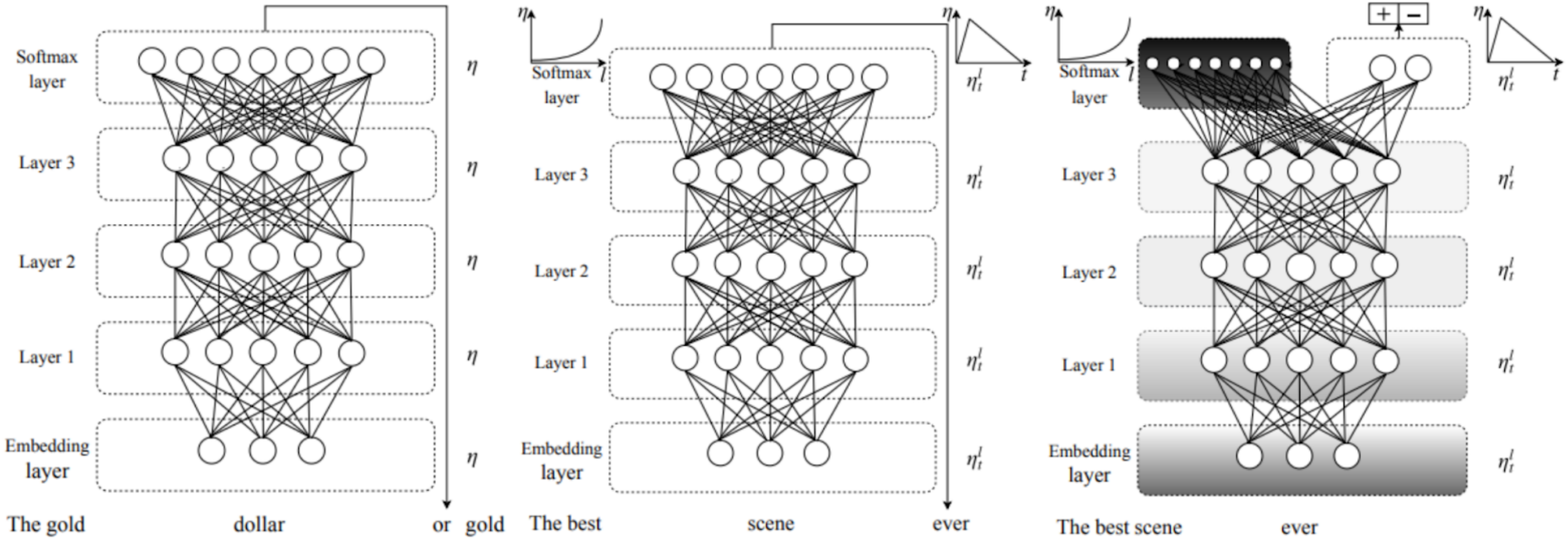


Methodology

Transfer Learning



UlmFit
Architecture

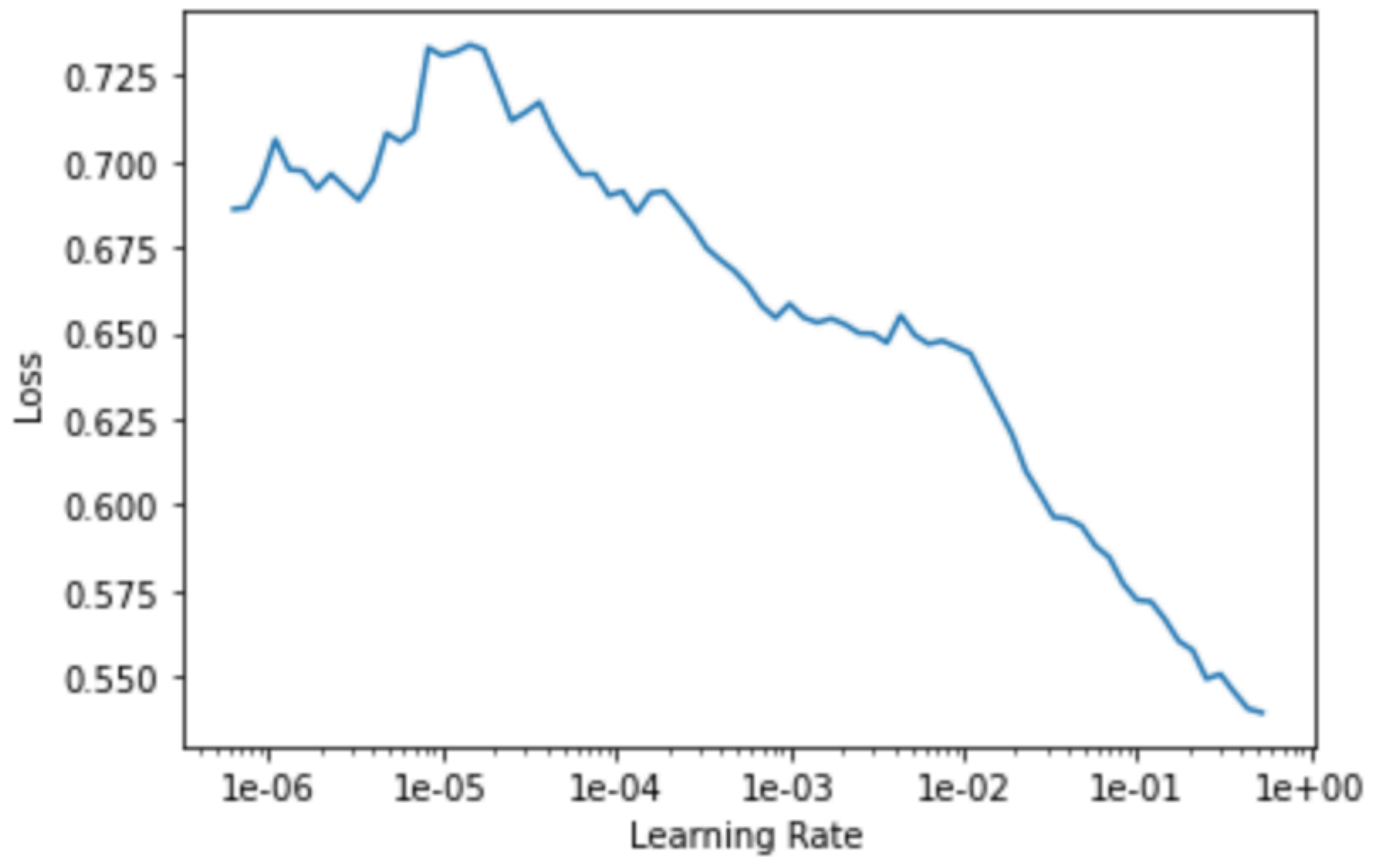


(a) LM pre-training (b) LM fine-tuning (c) Classifier fine-tuning

UlmFit

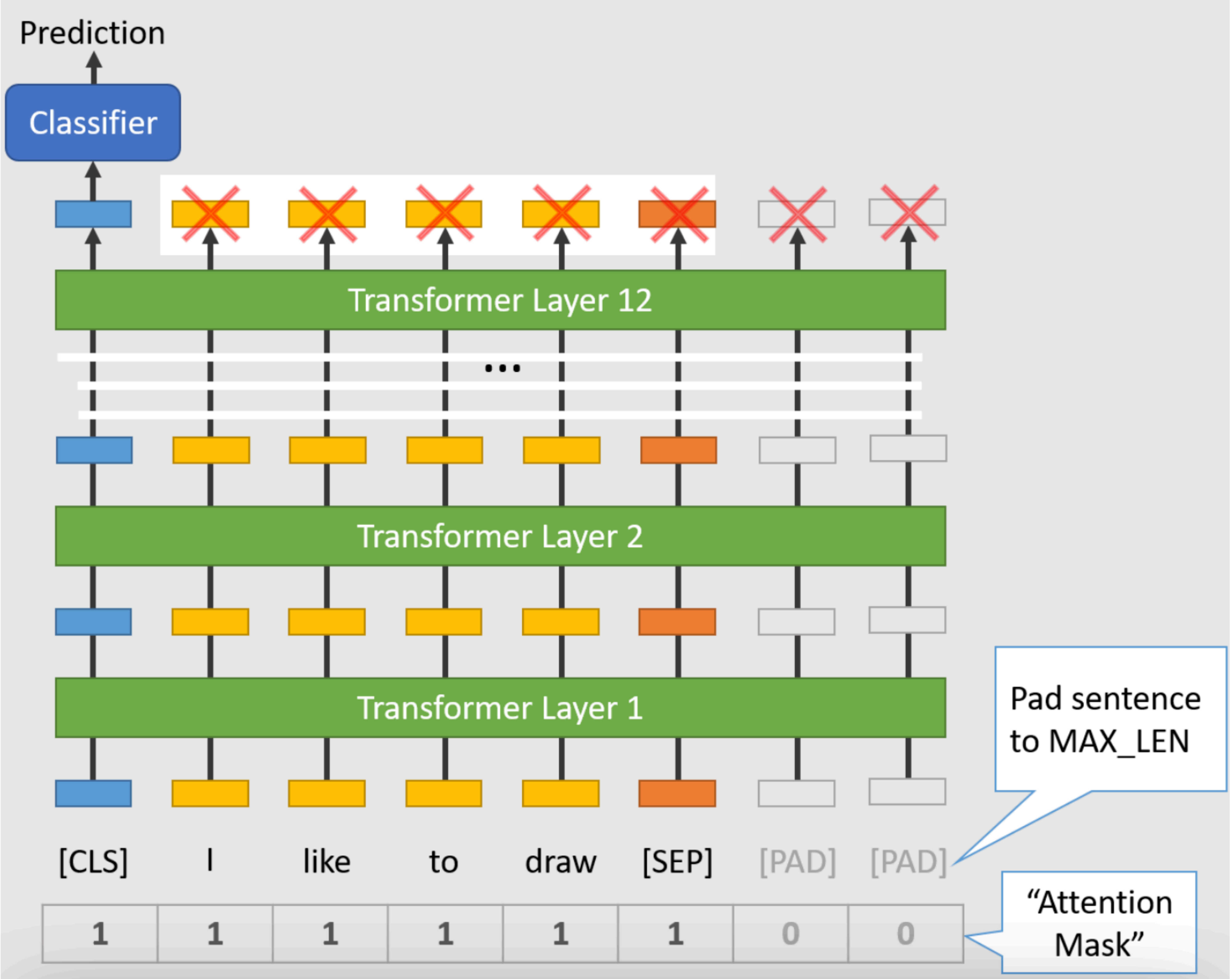
Training

- We train the classifier carefully, by gradually unfreezing the last few layers and training a few layers at a time.
- We use mixed precision for greater speed, smaller memory footprint, and a regularizing effect.
- We adjust the value of dropout among layers. A larger dropout helps prevent overfitting.
- We find the optimal learning rate for training by using the inbuilt function to find the value at which the loss is the least.
- We use 'Discriminative learning rates', so that the last layer has a smaller learning rate than the initial layers as it requires more learning.
- We use the Adam Optimiser as it has proven to work well in many NLP related tasks.



BERT

Architecture



BERT
Layers

The BERT model has 201 different named parameters.

==== Embedding Layer ====

bert.embeddings.word_embeddings.weight	(30522, 768)
bert.embeddings.position_embeddings.weight	(512, 768)
bert.embeddings.token_type_embeddings.weight	(2, 768)
bert.embeddings.LayerNorm.weight	(768,)
bert.embeddings.LayerNorm.bias	(768,)

==== Output Layer ====

bert.pooler.dense.weight	(768, 768)
bert.pooler.dense.bias	(768,)
classifier.weight	(2150, 768)
classifier.bias	(2150,)

Added

==== First Transformer ====

bert.encoder.layer.0.attention.self.query.weight	(768, 768)
bert.encoder.layer.0.attention.self.query.bias	(768,)
bert.encoder.layer.0.attention.self.key.weight	(768, 768)
bert.encoder.layer.0.attention.self.key.bias	(768,)
bert.encoder.layer.0.attention.self.value.weight	(768, 768)
bert.encoder.layer.0.attention.self.value.bias	(768,)
bert.encoder.layer.0.attention.output.dense.weight	(768, 768)
bert.encoder.layer.0.attention.output.dense.bias	(768,)
bert.encoder.layer.0.attention.output.LayerNorm.weight	(768,)
bert.encoder.layer.0.attention.output.LayerNorm.bias	(768,)
bert.encoder.layer.0.intermediate.dense.weight	(3072, 768)
bert.encoder.layer.0.intermediate.dense.bias	(3072,)
bert.encoder.layer.0.output.dense.weight	(768, 3072)
bert.encoder.layer.0.output.dense.bias	(768,)
bert.encoder.layer.0.output.LayerNorm.weight	(768,)
bert.encoder.layer.0.output.LayerNorm.bias	(768,)

BERT
Training

- BERT pre trained model (from Hugging face library (Pytorch)
- Random batch sampling
- learning scheduler (learning rate= 2e-5 , epochs=4)
- Trained WordPiece tokenizer (cased version)
- Adam Optimizer
- Cleared gradients to make training efficient..

Results

Baseline accuracies on our dataset

	Naive Bayes	SVM's	Random Forests	AAAI-21
Feature vector representations	0.56	0.49	0.37	0.72
GloVe embeddings	0.62	0.54	0.44	

Results using transfer learning

UlmFit	BERT
0.71	0.85

Conclusion

- We observe that the BERT pre-trained model provides the best accuracy of 0.85 on our dataset.
- Its performance can be attributed to the fact that BERT applies the bidirectional training of Transformer to language modelling.
- Transformers don’t suffer from long dependency issues. Moreover, multi-head attention and positional embeddings both provide information about the relationship between different words.
- UlmFit uses the LSTM architecture underneath. Some of the issues with LSTM’s are,
- LSTM’s cannot be trained in parallel.
- They ‘forget’ the information regarding a word after a few time steps.
- Transformers are better than all the other architectures because they process sentences as a whole and learn relationships between words thank's to multi-head attention mechanisms and positional embeddings.

Future Work

- Use other pretrained models like RoBerTa, ERNIE, T5, etc for the task and perform a comparative study of there performance on the dataset.
- Test our approach on other standard datasets.

References

- http://neuralnetworksanddeeplearning.com/chap1.html -
- https://github.com/aadarshsingh191198/AAAI-21-SDU-shared-task-1-AI -
- https://link.springer.com/article/10.1007/s10472-018-9608-8
- https://www.aclweb.org/anthology/W09-1309.pdf
- https://cs.nyu.edu/media/publications/TR2015-973.pdf
- https://www.aclweb.org/anthology/P18-1121.pdf
- https://dl.acm.org/doi/10.5555/2888116.2888316
- https://www.sciencedirect.com/science/article/pii/S1532046418301552 -
- https://arxiv.org/pdf/1711.09271.pdf -
- https://arxiv.org/pdf/1910.14076.pdf