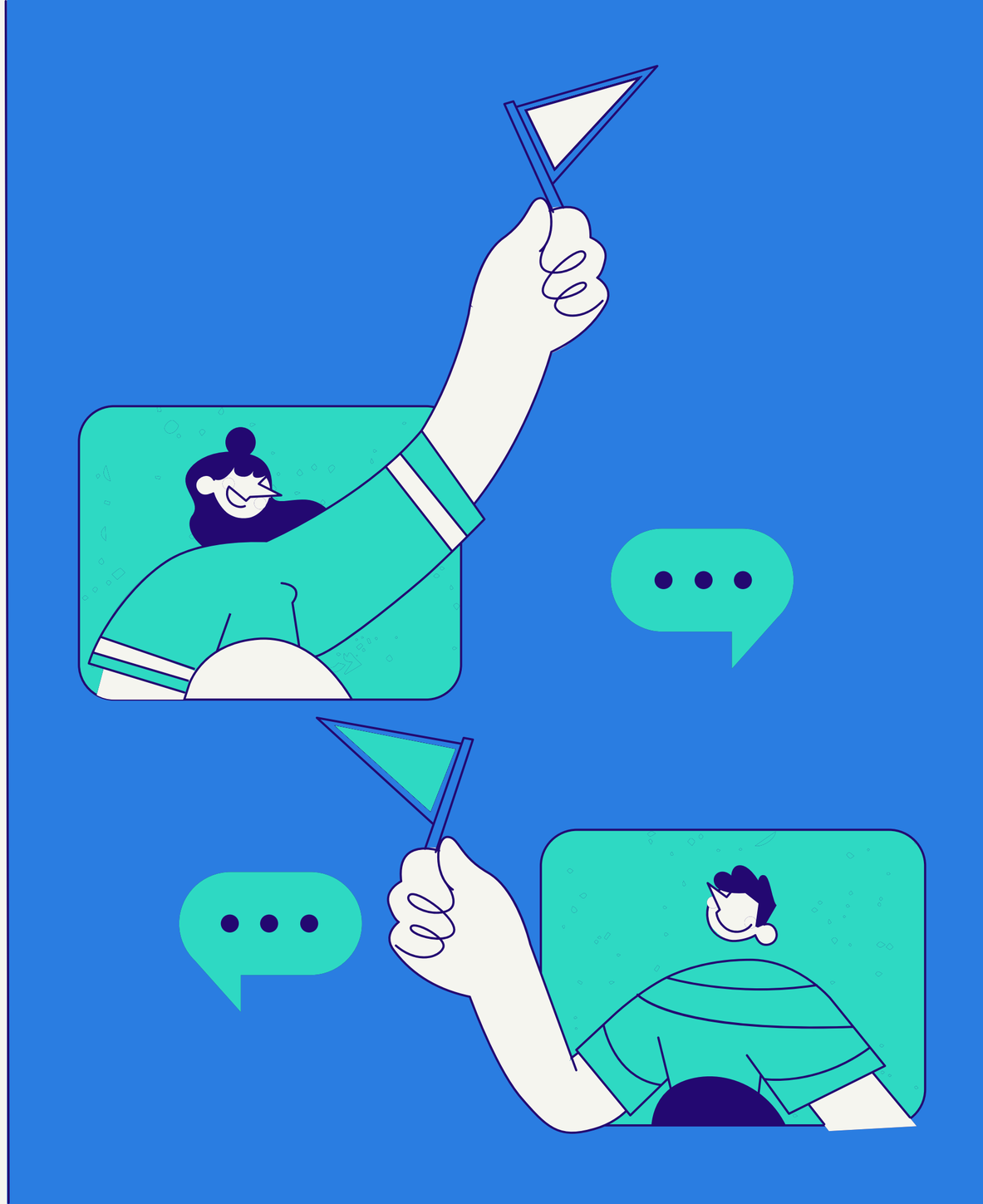
Question Answering System

Group Member: Wade Wei, Tong Zhou
Theme: Education & knowledge management



Agenda

Topics Covered

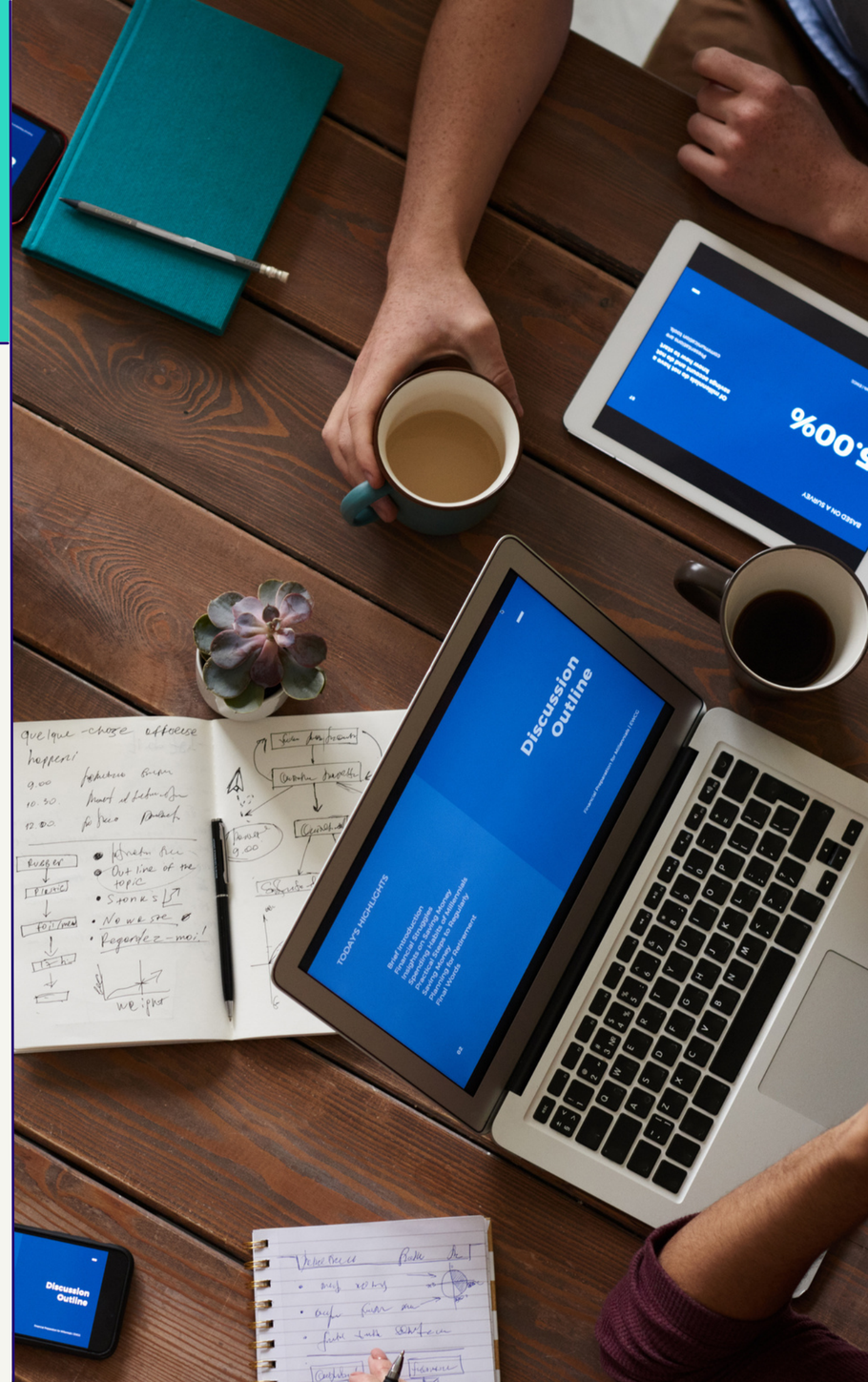
1. Background
2. Dataset
3. Models
4. Demo

Background

- Question Answering is the task of answering questions (typically reading comprehension questions).

- The main objective of our project is to create an NLP system that provides a single answer, which is obtained by selecting a span of text from the paragraph.

context + question = answer



Dataset - Subjqa

SubjQA is a question answering dataset that focuses on subjective questions and answers.

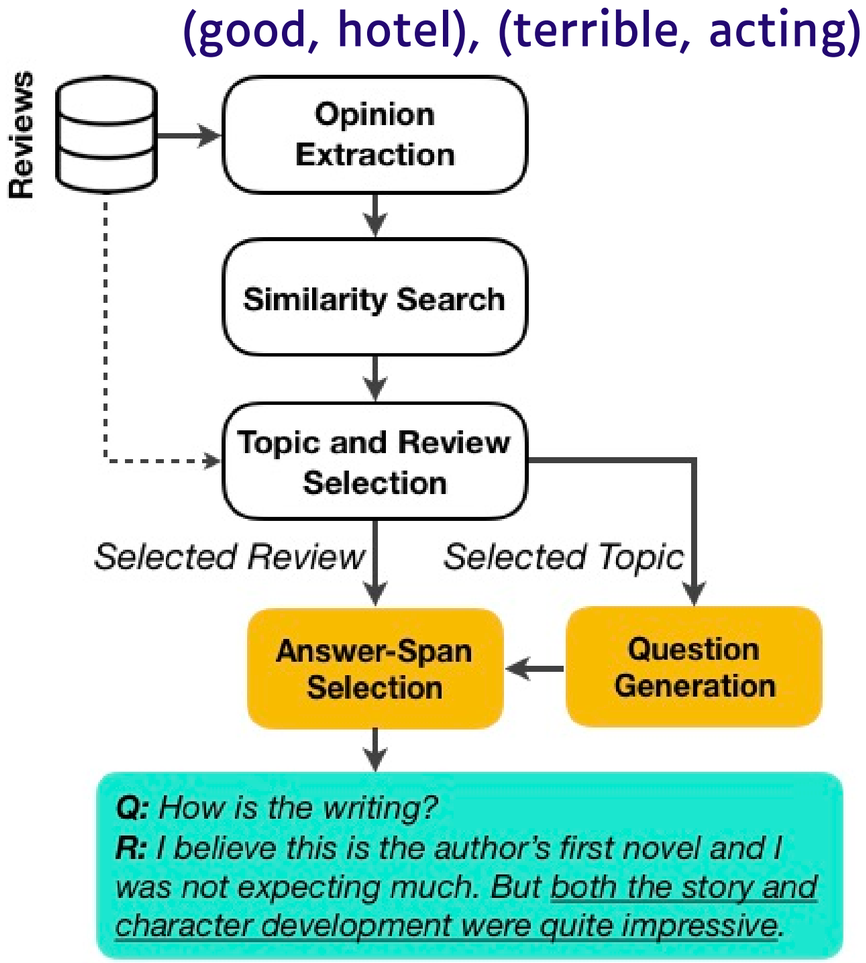
eg.

- factual question: "How much does this product weigh?"
- subjective question: "Is this easy to use?"

Dataset - Subjqa

Domain	Train	Dev	Test	Total
TripAdvisor	1165	230	512	1686
Restaurants	1400	267	266	1683
Movies	1369	261	291	1677
Books	1314	256	345	1668
Electronics	1295	255	358	1659
Grocery	1124	218	591	1725

Dataset - Subjqa



"responsive keys"

| (implied)

"good keyboard"

| (generate question)

"Is this keyboard any good?"

| (from review)

correct answer span

Dataset - Subjqa


```
{
    "answers": {
        "ans_subj_score": [1.0],
        "answer_start": [324],
        "answer_subj_level": [2],
        "is_ans_subjective": [true],
        "text": ["This is a wonderfully written book"],
    },
    "context": "While I would not recommend this book to a
    "domain": "books",
    "id": "0255768496a256c5ed7caed9d4e47e4c",
    "is_ques_subjective": false,
    "nn_asp": "matter",
    "nn_mod": "interesting",
    "q_reviews_id": "a907837bafe847039c8da374a144bff9",
    "query_asp": "part",
    "query_mod": "fascinating",
    "ques_subj_score": 0.0,
    "question": "What are the parts like?",
    "question_subj_level": 2,
    "review_id": "a7f1a2503eac2580a0ebbc1d24fffca1",
    "title": "0002007770",
}
```


context

Story is intriguing though incredulous. But I felt the book dragged on a bit too long. I did not feel compelled to any of the characters.

question

What do you think about the story?

true answer

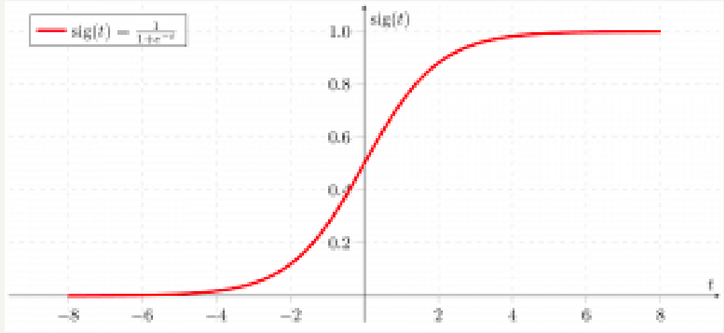
Story is intriguing though incredulous.

Data Processing

There are some answers missing, so we need to filter out those to achieve a better performance

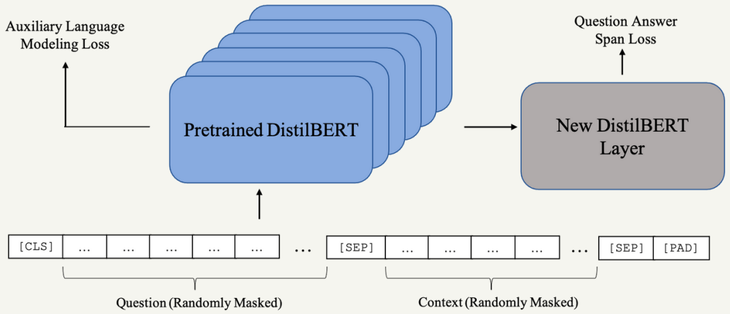
```
train_dataset = train_dataset.filter(lambda example: len(example["answers"]["text"])>0)
val_dataset = val_dataset.filter(lambda example: len(example["answers"]["text"])>0)
test_dataset = test_dataset.filter(lambda example: len(example["answers"]["text"])>0)
```

Model



Non-deep Learning

Logistic Regression



Deep Learning

DistilBERT

Non-deep learning Model

Create Embedding from context

Compute cosine similarity and euclidean distance with question

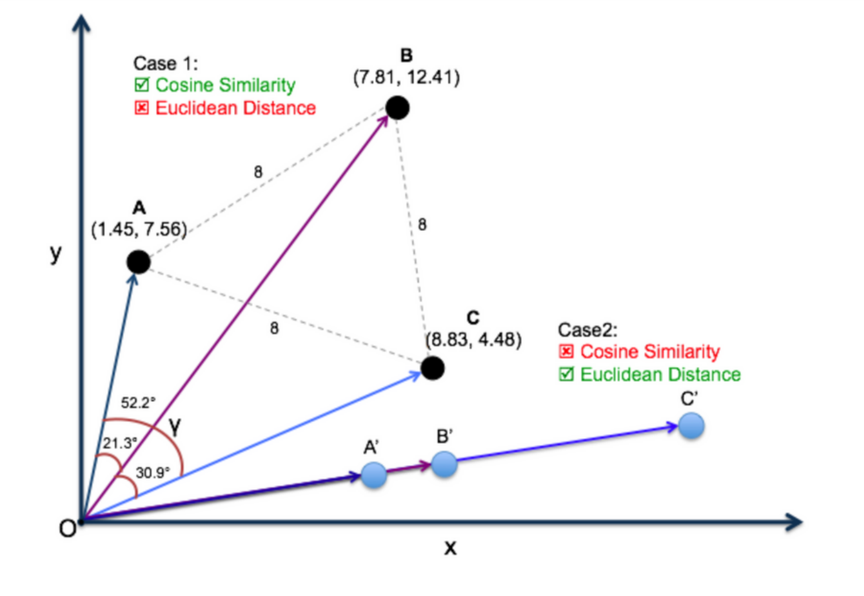
Create feature and label

Perform logistic regression

nltk package to tokenize a context into sentences

```
context:  I loved the movie, but the book is so much better.The ending is so perfect! I wish it was longer!
tokenized context:  ['I loved the movie, but the book is so much better.The ending is so perfect!', 'I wish it was longer!']
```

Two metrics to create feature: Cosine Similarity, Euclidean Distance



Cosine similarity features	Euclidean Distance features	label
50	50	1

	euc_dis	cos_sim	target
0	[1.3848074674606323, 1.3195196390151978, 1.380...	[0.041154082864522934, 0.12943391501903534, 0....	3
1	[1.3966732025146484, 1.4182605743408203, 1.312...	[0.024651864543557167, -0.00573156401515007, 0...	2
2	[1.2623049020767212, 1.3926451206207275]	[0.20329320430755615, 0.030269725248217583]	0
3	[1.2606918811798096, 1.2801878452301025, 1.368...	[0.20532794296741486, 0.18055947124958038, 0.0...	5
4	[1.1620548963546753, 1.2187520265579224, 1.361...	[0.3248140215873718, 0.25732165575027466, 0.07...	0

logistic regression, use 100 features and 1 label

For logistic regression, multinomial for multiple classes,
and newton-cg with l2 penalty
final accuracy: 0.31

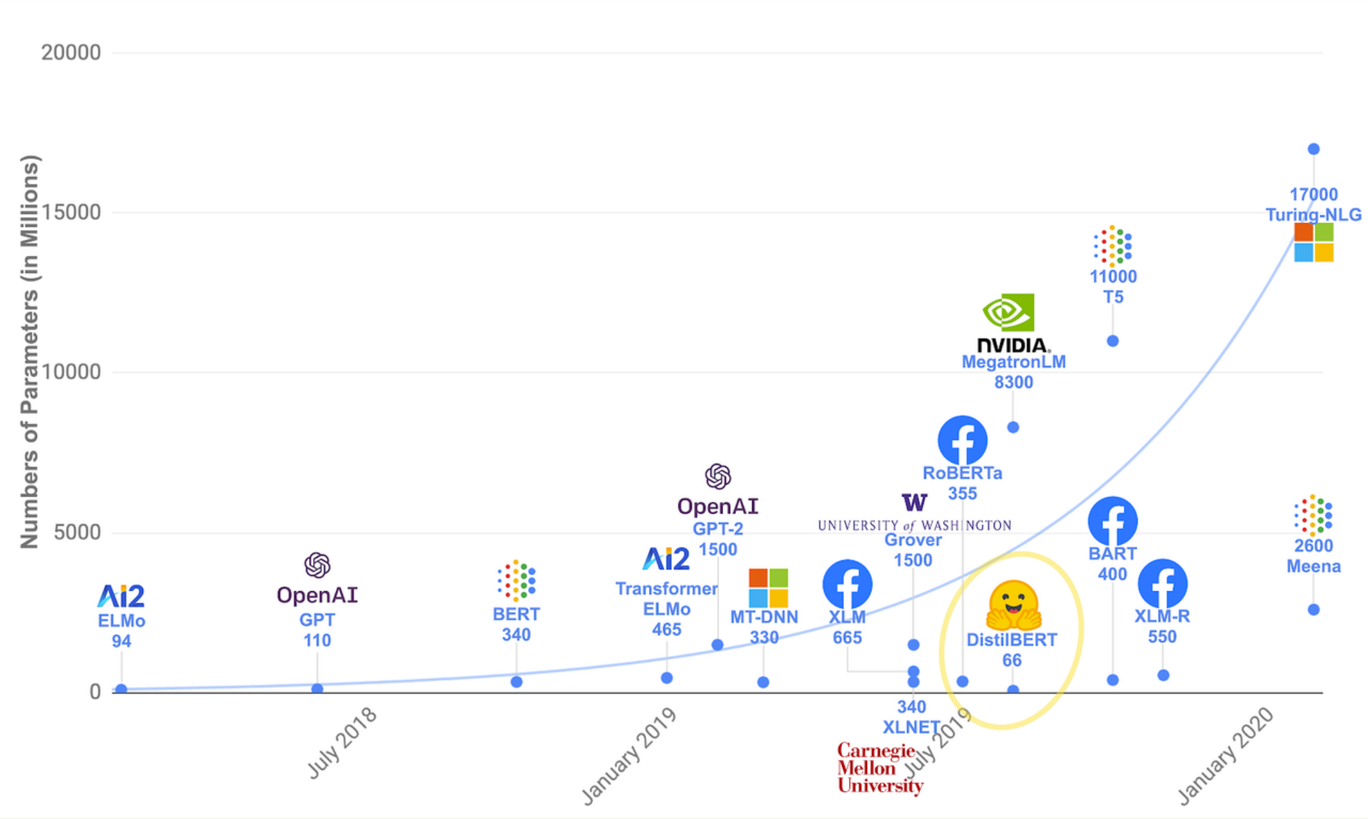
```
mul_lr = linear_model.LogisticRegression(multi_class='multinomial', solver='newton-cg')
mul_lr.fit(train_x, train_y)

print("Multinomial Logistic regression Train Accuracy : ", metrics.accuracy_score(train_y, mul_lr.predict(train_x)))
print("Multinomial Logistic regression Test Accuracy : ", metrics.accuracy_score(test_y, mul_lr.predict(test_x)))
```

```
Multinomial Logistic regression Train Accuracy :  0.3438106325184323
Multinomial Logistic regression Test Accuracy :  0.31317829457364343
```

Deep learning Model

DistilBERT is a small, fast, cheap and light Transformer model trained by distilling BERT base.

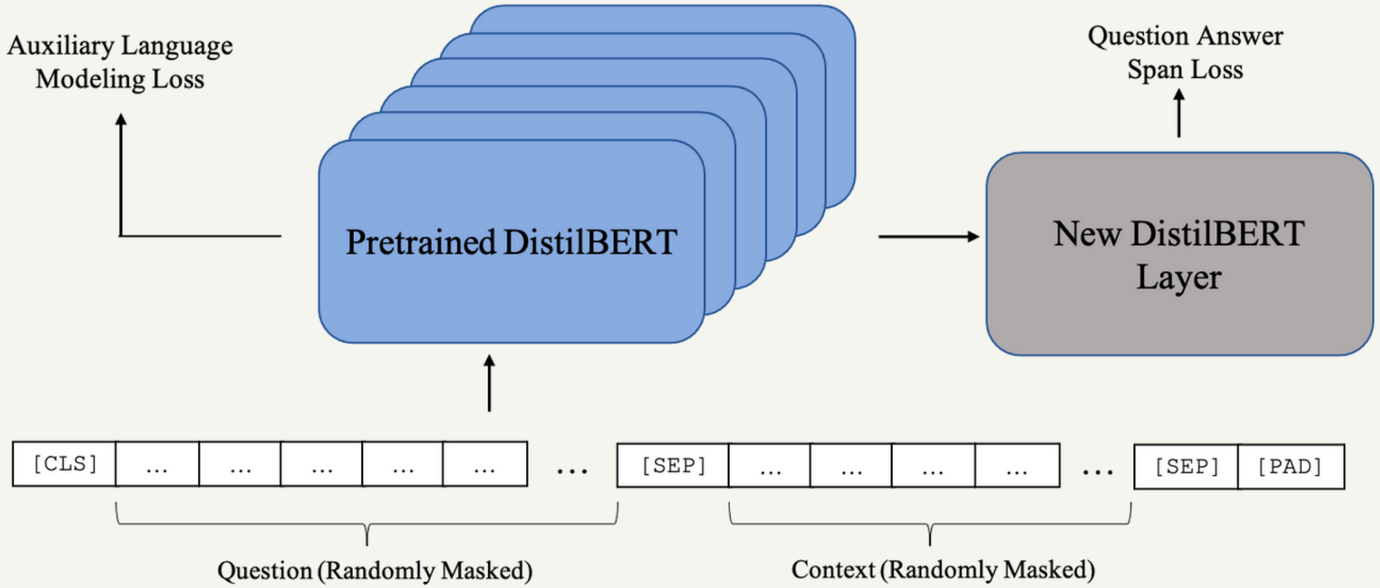


DistilBERT: distillation loss, simpler structure

Training loss The student is trained with a distillation loss over the soft target probabilities of the teacher: $L_{ce} = \sum_i t_i * \log(s_i)$ where t_i (resp. s_i) is a probability estimated by the teacher (resp. the student). This objective results in a rich training signal by leveraging the full teacher distribution. Following Hinton et al. [2015] we used a *softmax-temperature*: $p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$

The token-type embeddings and the pooler are removed while the number of layers is reduced by a factor of 2.

Linear Algebra wise, many operations are highly optimized - think of for a matrix, it becomes convertible



Final layer is replaced to achieve QA task

not computing a probability distribution over answers, but two probability distributions over the tokens in the document text, representing the start and end of the span containing the answer.

Training

Parameters Optimizer: Adam
lr: 5e-5
The maximum length of a feature
(question and context): 384
The authorized overlap between two
part of the context when splitting: 128

```
Epoch 1/3
147/147 [==============================] - 121s 821ms/step - loss: 2.1127 - val_loss: 2.3692
Epoch 2/3
147/147 [==============================] - 121s 821ms/step - loss: 1.6078 - val_loss: 2.6038
Epoch 3/3
147/147 [==============================] - 121s 821ms/step - loss: 1.0398 - val_loss: 3.1525
<keras.callbacks.History at 0x7fc93f7b1650>
```

Metrics

If predicted answer is a substring of true answer, or vice versa --> TP

If not substring and vice versa --> FP

If predicted answer is ' ' --> FN

No TN because all my dataset have answers (all P)

```
  if true_answer in predicted_answer or predicted_answer in true_answer:
    TP += 1
  if len(predicted_answer) < 1:
    FN += 1
  if true_answer not in predicted_answer and predicted_answer not in true_answer:
    FP += 1

  #print(TP, FP, P)

print('DistilBERT test sensitivity ', TP/(TP + FN))
print('DistilBERT test F1 score ', 2 * TP/(2*TP + FP + FN))
```

```
DistilBERT val sensitivity  0.6854990583804144
DistilBERT val F1 score  0.6179966044142614

DistilBERT test sensitivity  0.7015457788347206
DistilBERT test F1 score  0.6413043478260869
```

Comparison

Higher accuracy when using NLP model

70% F1 and sensitivity on both validation set and test set, while only 30% for logistic regression

Easier to implement with logistic regression, require less resources

Only nltk to tokenize, and scikit to do regression; for NLP, cannot do training with GPU

Summary

How to improve performance?

- Hyperparameter tuning
- Cleaner dataset
- Other models with BERT base

Further Work

- Expand dataset
- Compare different models
- play with features to do more non NLP could be interesting as well

[Demo]

- From streamlit.

Streamlit

context

Story is intriguing though incredulous. But I felt the book dragged on a bit too long. I did not feel compelled to any of the characters.

question

What do you think about the story?

context

Story is intriguing though incredulous. But I felt the book dragged on a bit too long. I did not feel compelled to any of the characters.

question

What do you think about the story?

true answer

Story is intriguing though incredulous.

Thank you!