浙江大学管理学院
SCHOOL OF MANAGEMENT
ZHEJIANG UNIVERSITY

Text mining and classifications in Yunduoduo

Team members: Wang Jiayun ZhangNing

December, 2021

Data Source:
Zheda Yunduoduo

> 朵朵校友圈 is a new type of "open source" campus community that is rapidly becoming popular among college students and is self-managed and self-served by students. We focus on local users centered on each college and university, and are committed to providing an efficient and easy-to-use mutual welfare platform and a harmonious social media to help teachers and students solve the difficulties in study, life and work.

Outline

- BERT Model introduction
- Data description
- Sentiment and trend analysis
- Wordcloud analysis
- Automatic topic classification system

BERT

1. What is BERT?

- BERT (Bidirectional Encoder Representations from Transformers) is awesome research in Natural Language Processing (NLP) published **by researchers at Google AI Language in 2018**.
- It has caused a stir in the Machine Learning community by presenting state-of-the-art results in a wide variety of NLP tasks, including Neural Machine Translation, Question Answering (SQuAD v1.1), Sentence Pair Classification task (MNLI), Sentiment Analysis, Text Summarization and others.

① It is Bi-directional

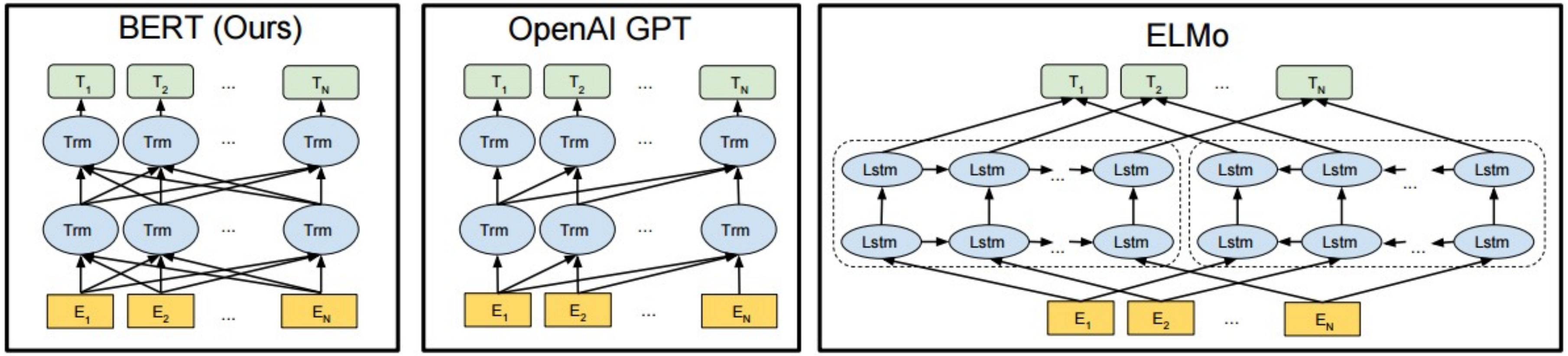
② It uses an Encoder Representation

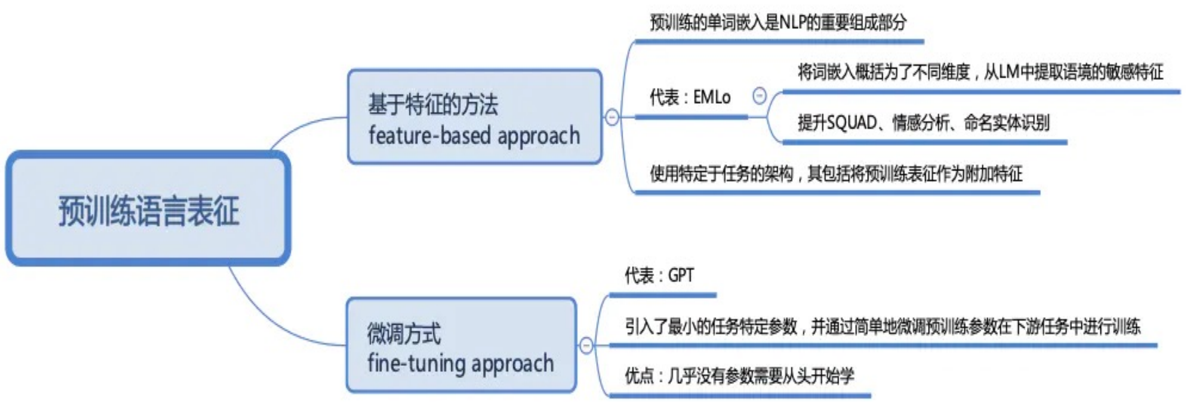
③ It has a Transformer based architecture

- **→BERT's key technical innovation is applying the bidirectional training of Transformer, a popular attention model, to language modeling**. This is in contrast to previous efforts which looked at a text sequence either from left to right or combined left-to-right and right-to-left training.
- A language model that is bi-directionally trained can have **a deeper sense of language context and flow** than single-direction language models.
- In the paper, the researchers detail a novel technique named **Masked Language Modelling (MLM)** which allows bidirectional training in models in which it was previously impossible.

Model Comparison

——How BERT comes and why it becomes so popular ?





模型	获得长距离语义信息程度	左右上下文语义	是否可以并行
Word2vec	1	True	True
单向LSTM	2	False	False
ELMo	2	True	False
BERT	**3**	**True**	**True**

1. From Word to Vectors

Tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as punctuation.

1.Using **wordpieces** (e.g. playing -> play + ##ing) instead of words.

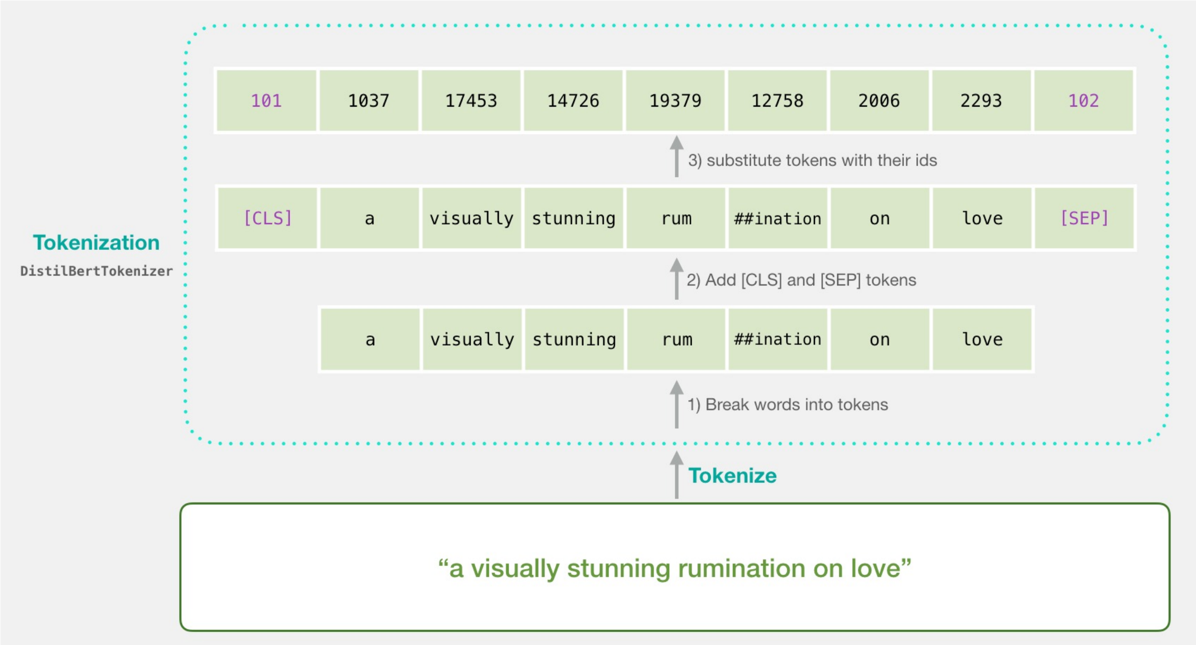
2.Numericalization aims at mapping each token to a unique integer in the corpus' vocabulary.

3.Token embedding is the task of get the embedding (i.e. a vector of real numbers) for each word in the sequence. Each word of the sequence is mapped to a emb_dim dimensional vector that the model will learn during training.

4.Padding was used to make the input sequences in a batch have the same length.

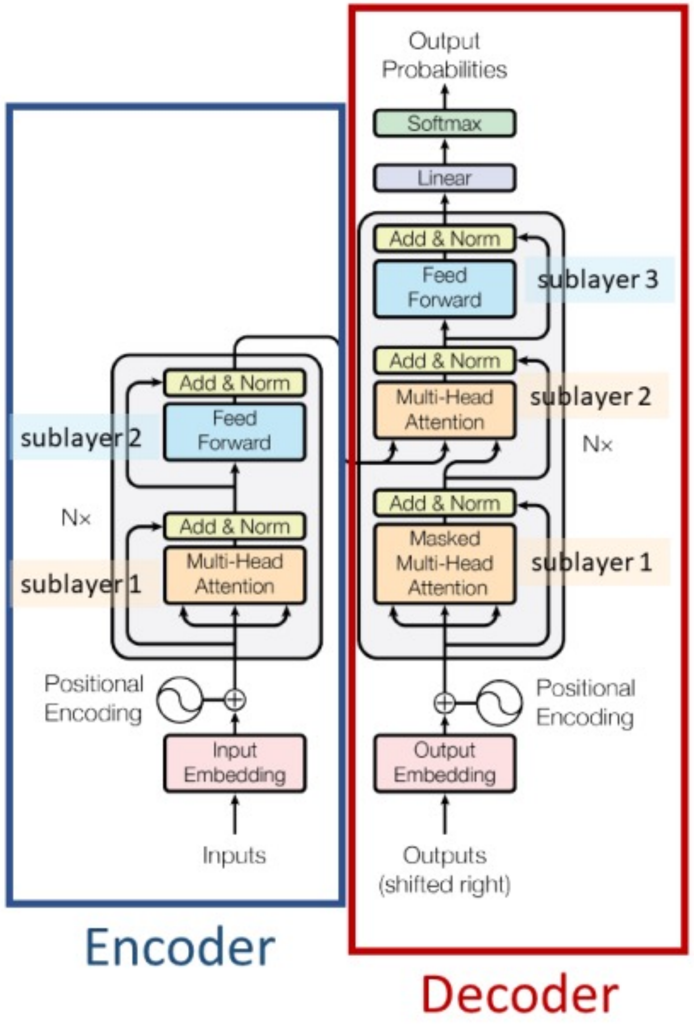
5.Positional encoding is designed to help the model learn some notion of sequences and relative positioning of tokens.

6.Sentence embedding techniques represent entire sentences and their semantic information as vectors.



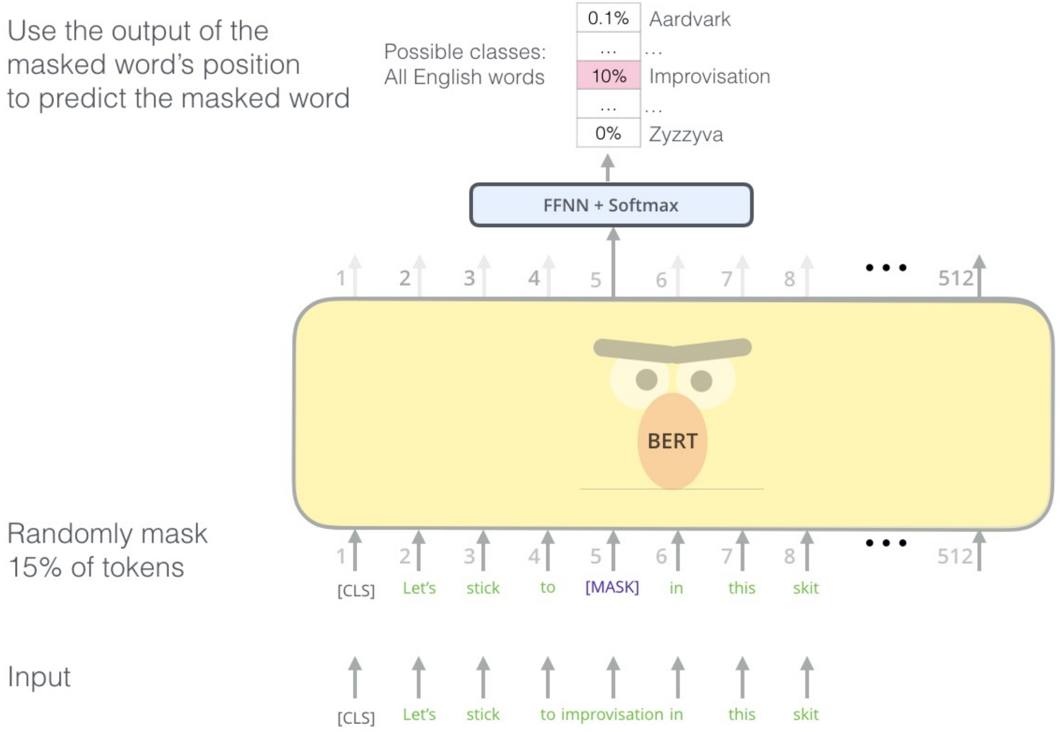
2. The Encoders from the transformer

- The transformer architecture is based on encoder-decoder form.
- The encoder consists of four parts (**self attention, multi-head attention, residual connections and normalization and feed forward network**). The encoder of the transformer architecture looks like :
  - multi head attention--measure the self attention score multiple times
  - Add and Norm--Add means the residual connection and norm mean the layer normalization. It also use dropout in this layer to reduce overfitting.



3. Masked Language Modeling (MLM)

- One of the greatest feature of BERT is the Masked Language Modeling (MLM).
- **MLM randomly masks some of the tokens from the input,** and the objective is to **predict the original vocabulary id of the masked word** based only on its context. Unlike left-to-right language model pre-training, the MLM objective allows the representation to fuse the left and the right context, which allows us to pre-train a deep bidirectional Transformer.

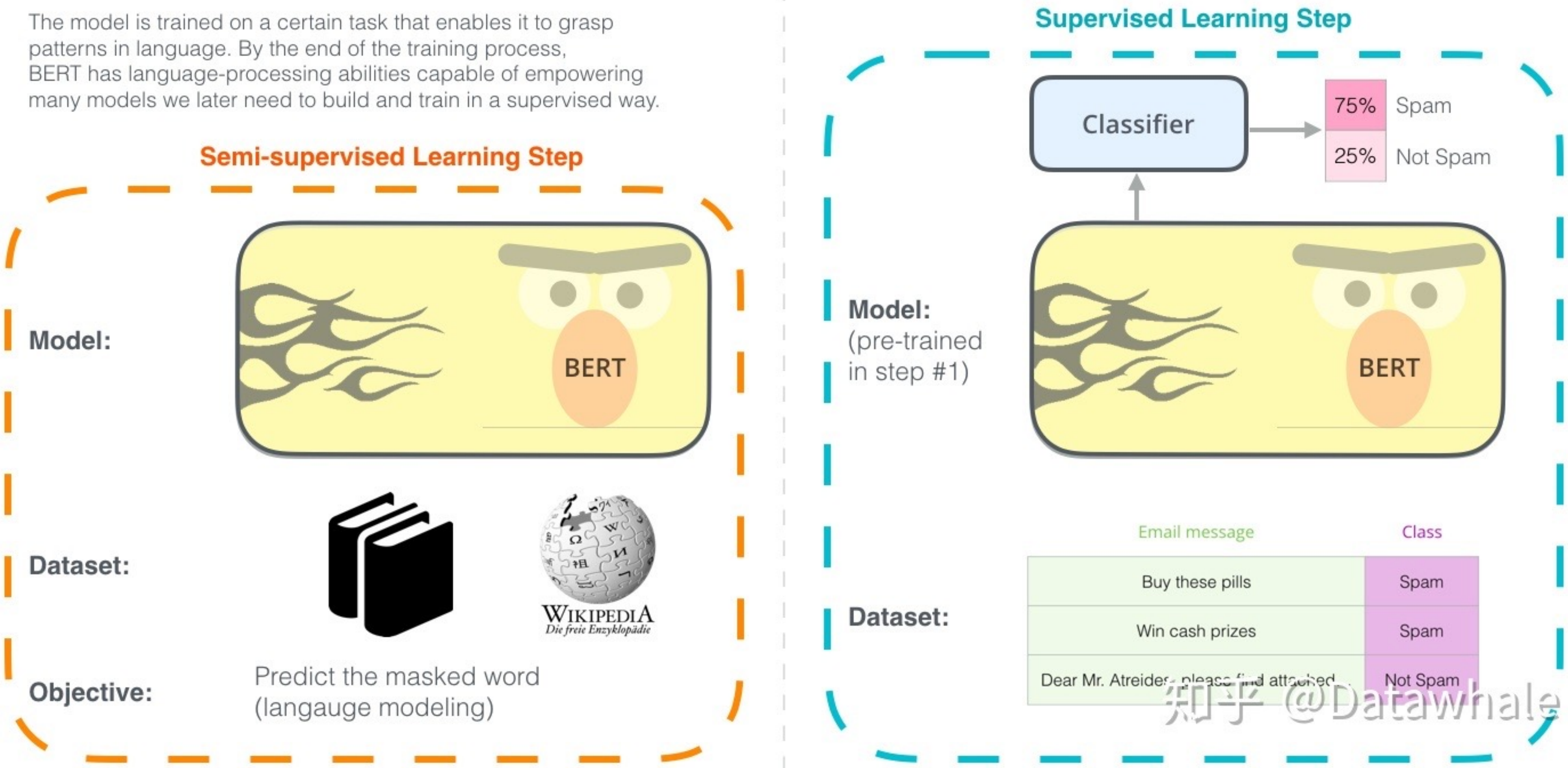


3. Masked Language Modeling (MLM)

1 - Semi-supervised training on large amounts of text (books, wikipedia..etc).

The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.

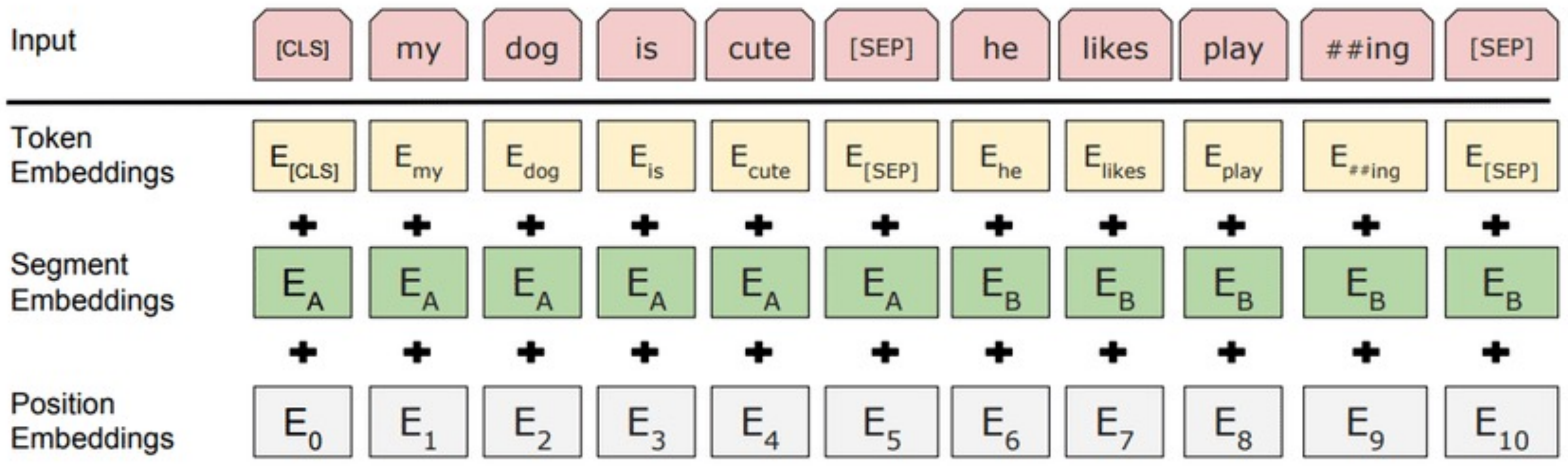
2 - Supervised training on a specific task with a labeled dataset.



4. Next Sentence Prediction (NSP)

In the BERT training process, the model receives pairs of sentences as input and learns to **predict if the second sentence in the pair is the subsequent sentence in the original document**. To help the model distinguish between the two sentences in training, the input is processed in the following way before entering the model:

- A **[CLS]** token is inserted at the beginning of the first sentence and a **[SEP]** token is inserted at the end of each sentence.
- A **sentence embedding** indicating Sentence A or Sentence B is added to each token. Sentence embeddings are similar in concept to token embeddings with a vocabulary of 2.
- A **positional embedding** is added to each token to indicate its position in the sequence.

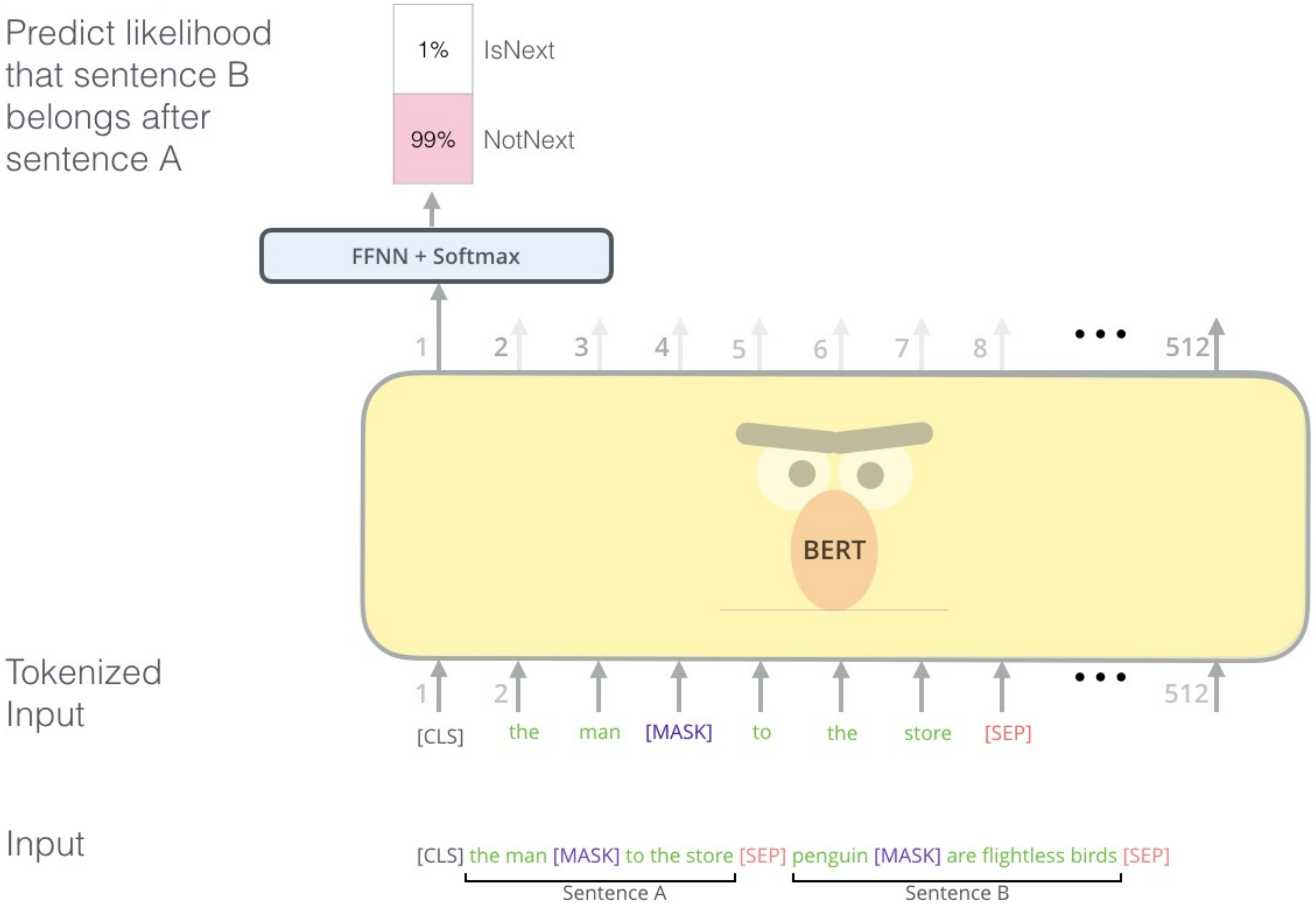


3/4: BERT pre-training method

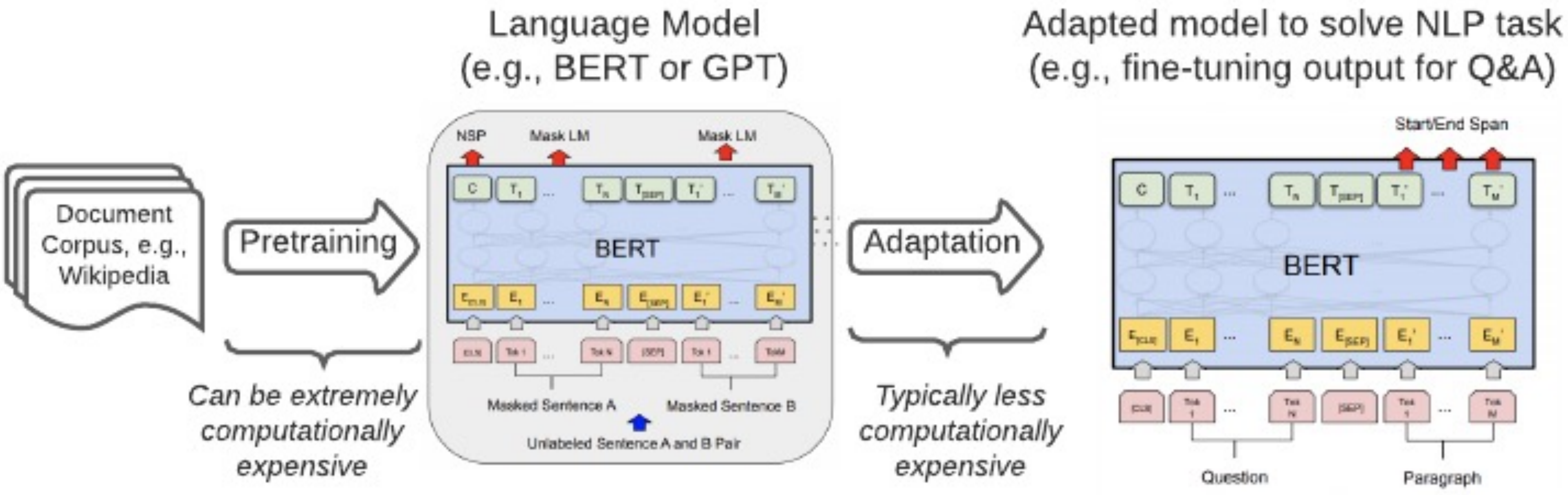
When the BERT model is pre-trained, it is based on 2 types of tasks:

- Masked language modeling (language model with mask)
- Next sentence prediction

When training the BERT model, Masked LM and Next Sentence Prediction are trained together, with the goal of minimizing the combined loss function of the two strategies. Now, the whole BERT model in this looks like:



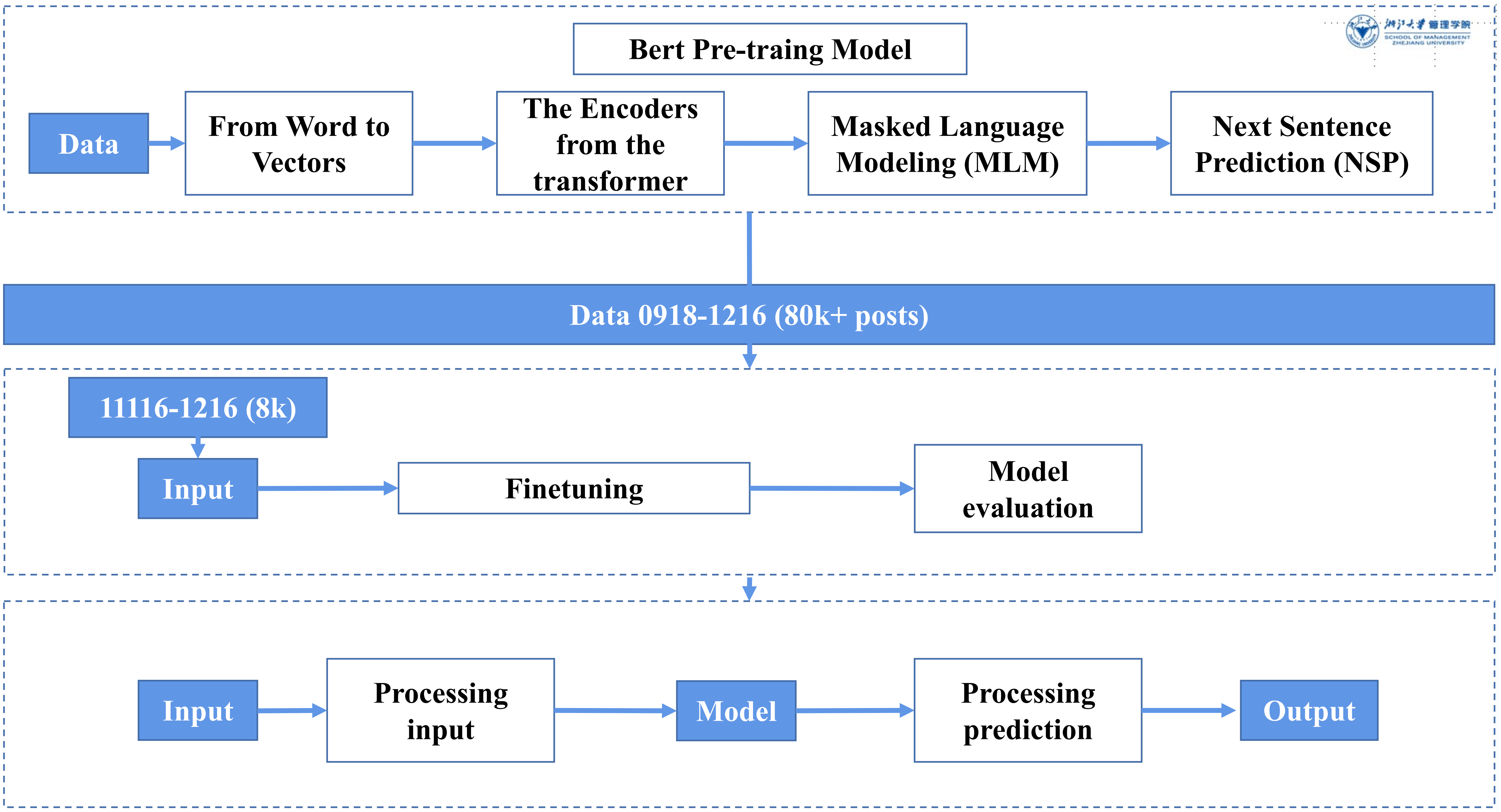
5. BERT as Transfer Learning in NLP



APPLICATIONS

A different variation of BERT is now using in different real-life projects. Models trained on domain/application-specific corpus are Pre-trained models. Training on domain-specific corpus has shown to yield better performance when fine-tuning them on downstream NLP tasks like NER etc. for those domains, in comparison to fine tuning BERT. Some of the variations are listed below that are using different real-world NLP problem

- RoBERta (robustly optimized BERT for solving different tasks)
- BioBERT (use for biomedical text)
- SciBERT (use for scientific publications)
- ClinicalBERT (use for clinical notes)
- G-BERT (use for medical/diagnostic code representation and recommendation)
- M-BERT from 104 languages for zero-shot cross-lingual model transfer (task-specific annotations in one language is used to fine-tune a model for evaluation in another language)
- ERNIE (knowledge graph) + ERNIE (2) incorporates knowledge into pre-training but by masking entities and phrases using KG.
- TransBERT — unsupervised, followed by two supervised steps, for a story ending prediction task
- videoBERT (a model that jointly learns video and language representation learning) by representing video frames as special descriptor tokens along with text for pretraining.



Bert Pre-traing Model
Data
From Word to Vectors
The Encoders from the transformer
Masked Language Modeling (MLM)
Next Sentence Prediction (NSP)
Data 0918-1216 (80k+ posts)
11116-1216 (8k)
Input
Finetuning
Model evaluation
Input
Processing input
Model
Processing prediction
Output

Finetuning

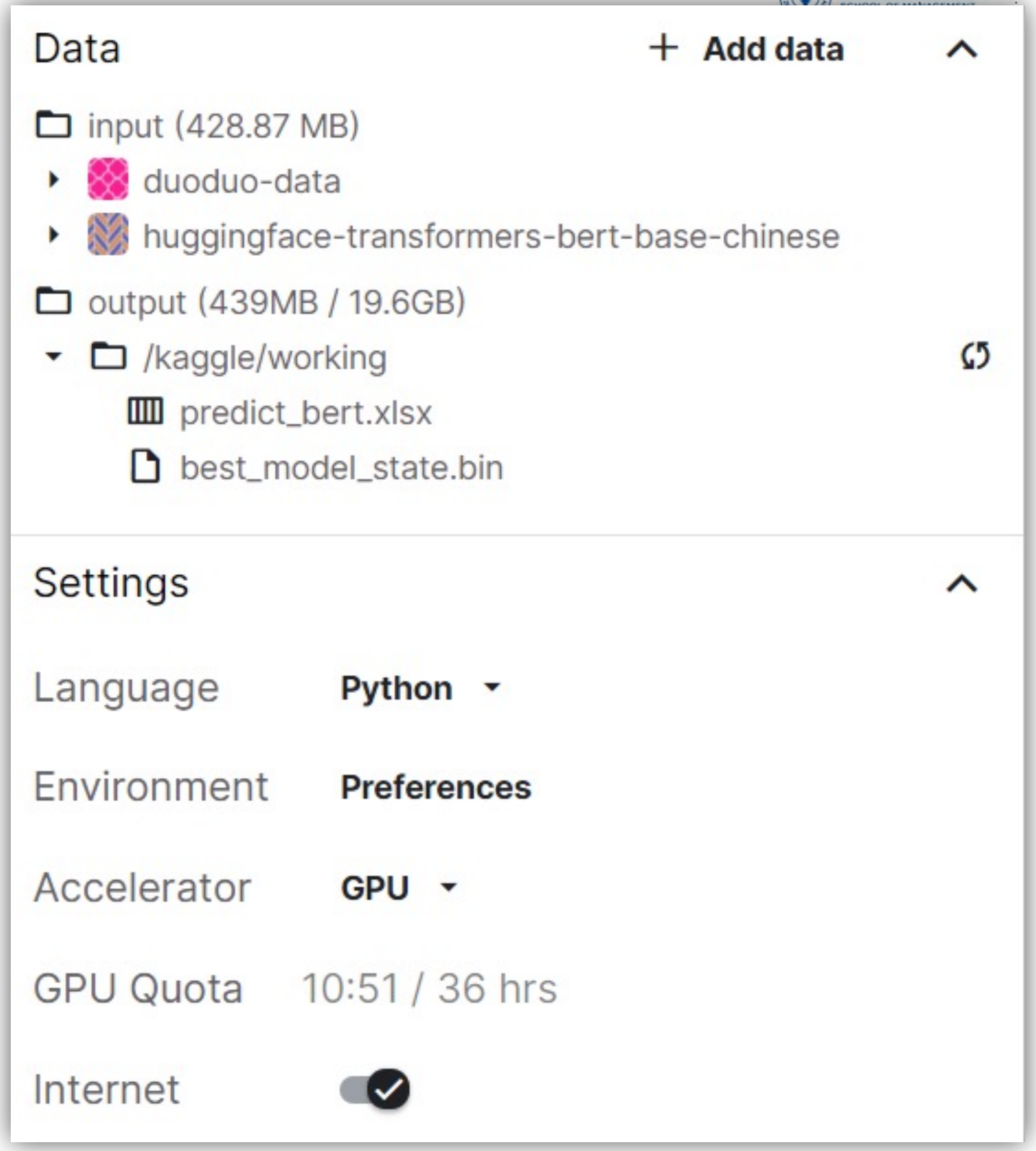
Specifically, we will take the pre-trained BERT model, add an untrained layer of neurons on the end, and train the new model for our classification task.

1. Setup

1.1. Using kaggle GPU for Training

We will train a large neural network, so we need hardware acceleration, otherwise the training will take a long time. Kaggle offers 36 hours a week's GPU for free, so we chose to program on the Kaggle platform.

1.2. Download chinese-roberta-wwm



Finetuning

2. Loading DuoDuo Dataset

2.1 Discription of dataset

- Our whole dataset contains data from 2021-09-18 —— 2021-12-16
- Filter : main post + “投稿”
- Our traning set contains data from 2021-11-16 —— 2021-12-16

  Amount of data : 7714

  The sentiment label Is labelled manually

```
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   comment_count    4379 non-null   float64
 1   text             7714 non-null   object
 2   create_time      4379 non-null   datetime64[ns]
 3   like_count       4379 non-null   float64
 4   post_id          4379 non-null   float64
 5   pre_context      0 non-null      float64
 6   pre_post_id      0 non-null      float64
 7   pre_user_id      0 non-null      float64
 8   root_post_id     4379 non-null   float64
 9   school           4379 non-null   object
 10  shida            4379 non-null   float64
 11  topic_1          4379 non-null   object
 12  topic_2          4379 non-null   object
 13  user_hot_value   4379 non-null   float64
 14  user_id          4379 non-null   object
 15  user_post_count  4379 non-null   float64
 16  sentiment_label  7714 non-null   int64
dtypes: datetime64[ns](1), float64(10), int64(1), object(5)
memory usage: 1.0+ MB
```

Finetuning

3. Tokenization & Input Formatting

In this section, we'll transform our dataset into the format that BERT can be trained on.

3.1. BERT Tokenizer

To feed our text to BERT, it must be split into tokens, and then these tokens must be mapped to their index in the tokenizer vocabulary.

```
tokenizer = BertTokenizer.from_pretrained('../input/huggingface-transformers-bert-base-chinese')
```

3.2. Required Formatting

The above code left out a few required formatting steps that we'll look at here.

We are required to:

1.Add special tokens to the start and end of each sentence.

2.Pad & truncate all sentences to a single constant length.

3.Explicitly differentiate real tokens from padding tokens with the "attention mask".

```
encoding=self.tokenizer.encode_plus(
    text,
    add_special_tokens=True,# 句子加CLS+text+SEP
    max_length=self.max_len,#
    return_token_type_ids=True,#两个句子之间分句 000111
    pad_to_max_length=True,#补全
    return_attention_mask=True,# 掩码 111000自注意机制
    return_tensors='pt',#pytorch类型
)
```

4. Train Our Classification Model

Now that our input data is properly formatted, it's time to fine tune the BERT model.

```
class EnterpriseDangerClassifier(nn.Module):
    def __init__(self, n_classes):
        super(EnterpriseDangerClassifier, self).__init__()
        self.bert = BertModel.from_pretrained('../input/huggingface-transformers-bert-base-chinese')
        self.drop = nn.Dropout(p=0.3)
        self.out = nn.Linear(self.bert.config.hidden_size, n_classes)
    def forward(self, input_ids, attention_mask):
        _, pooled_output = self.bert(
            input_ids=input_ids,
            attention_mask=attention_mask,
            return_dict = False
        )
#          max_pooling = max(pooled_output)
#          mean_pooling = mean(pooled_output)
        #out_put = cat(max_pooling, mean_pooling) #2*768 防止过拟合
        #output = self.drop(out_put)
        output = self.drop(pooled_output) # dropout
        return self.out(output)
```

Finetuning

4.1. Optimizer & Learning Rate Scheduler

Now that we have our model loaded we need to grab the training hyperparameters from within the stored model.

optimizer = AdamW

Loss function : CrossEntropyLoss

Epoch = 2

Leaning rate = 2e5

Batch size = 6

```
# 模型训练
EPOCHS = 2 # 训练轮数

optimizer = AdamW(model.parameters(), lr=2e-5, correct_bias=False)
total_steps = len(train_data_loader) * EPOCHS#总步长

scheduler = get_linear_schedule_with_warmup(
  optimizer,
  num_warmup_steps=0,
  num_training_steps=total_steps
)#调度器, 控制训练的步数,预热器,开始设置较小的学习率, 每次迭代增加一点, 直到一个相对较大的学习率,
#整个过程可以使模型更加稳定, 加快收敛速度

loss_fn = nn.CrossEntropyLoss().to(device)
```

Finetuning

4.2. Training Loop

Below is our training loop. There's a lot going on, but fundamentally for each pass in our loop we have a trianing phase and a validation phase.

Training:

•Unpack our data inputs and labels
•Load data onto the GPU for acceleration
•Clear out the gradients calculated in the previous pass.
  - In pytorch the gradients accumulate by default (useful for things like RNNs) unless you explicitly clear them out.

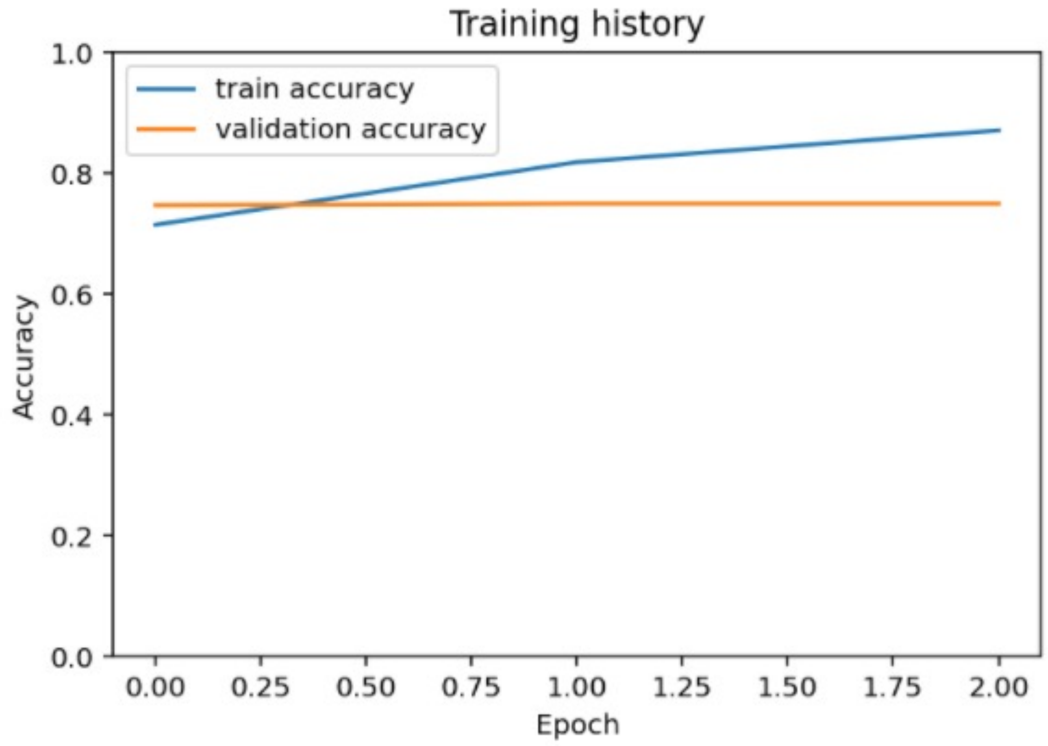
•Forward pass (feed input data through the network)
•Backward pass (backpropagation)
•Tell the network to update parameters with optimizer.step()
•Track variables for monitoring progress

Evalution:

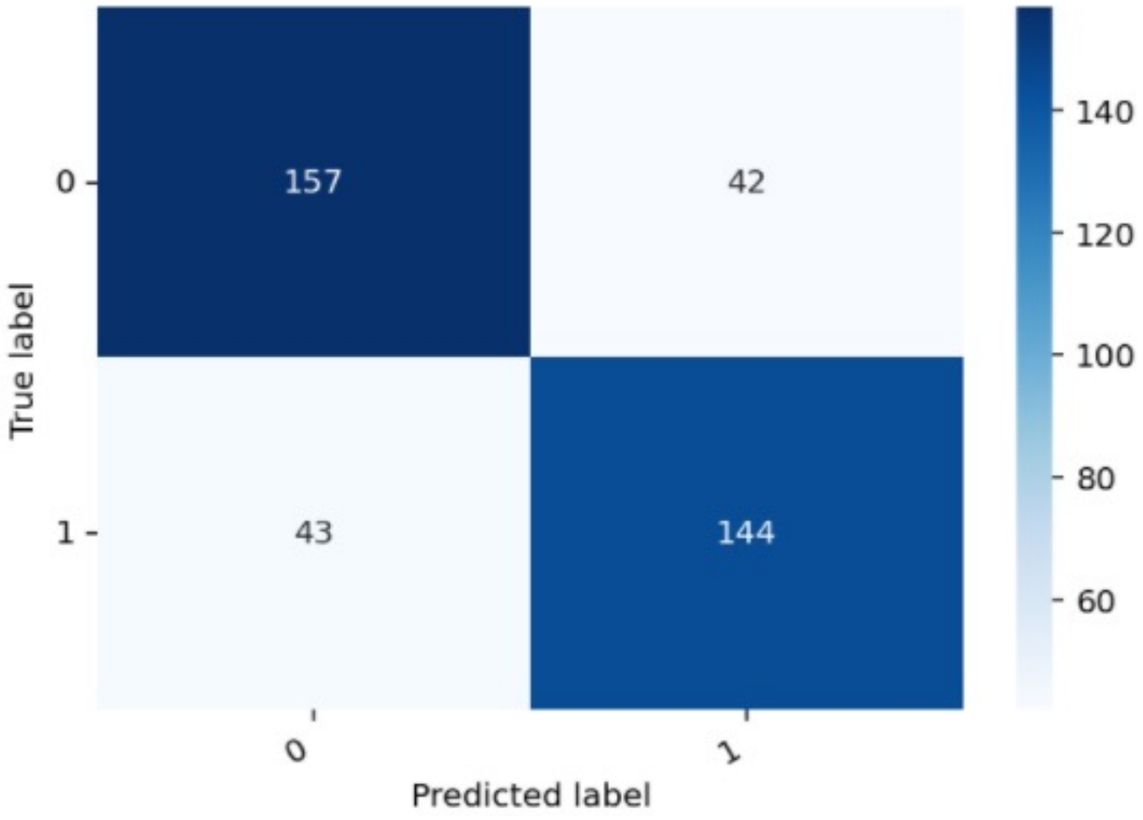
```
history = defaultdict(list) # 记录10轮loss和acc
best_accuracy = 0

for epoch in range(EPOCHS):

    print(f'Epoch {epoch + 1}/{EPOCHS}')
    print('-' * 10)

    train_acc, train_loss = train_epoch(
        model,
        train_data_loader,
        loss_fn,
        optimizer,
        device,
        scheduler,
        len(train)
    )

    print(f'Train loss {train_loss} accuracy {train_acc}')

    val_acc, val_loss = eval_model(
        model,
        val_data_loader,
        loss_fn,
        device,
        len(val)
    )

    print(f'Val   loss {val_loss} accuracy {val_acc}')
    print()

    history['train_acc'].append(train_acc)
    history['train_loss'].append(train_loss)
    history['val_acc'].append(val_acc)
    history['val_loss'].append(val_loss)

    if val_acc > best_accuracy:
        torch.save(model.state_dict(), 'best_model_state.bin')
```

Performance

```
Epoch 1/3
----------
Train loss 0.5898227646573412 accuracy 0.7136271967732641
Val   loss 0.537090925299204З accuracy 0.7461139896373057

Epoch 2/3
----------
Train loss 0.4674954005567777 accuracy 0.8173437049841544
Val   loss 0.6893748238969307 accuracy 0.7487046632124352

Epoch 3/3
----------
Train loss 0.4162559777750998 accuracy 0.8699222126188418
Val   loss 0.8516548585791427 accuracy 0.7487046632124352
```

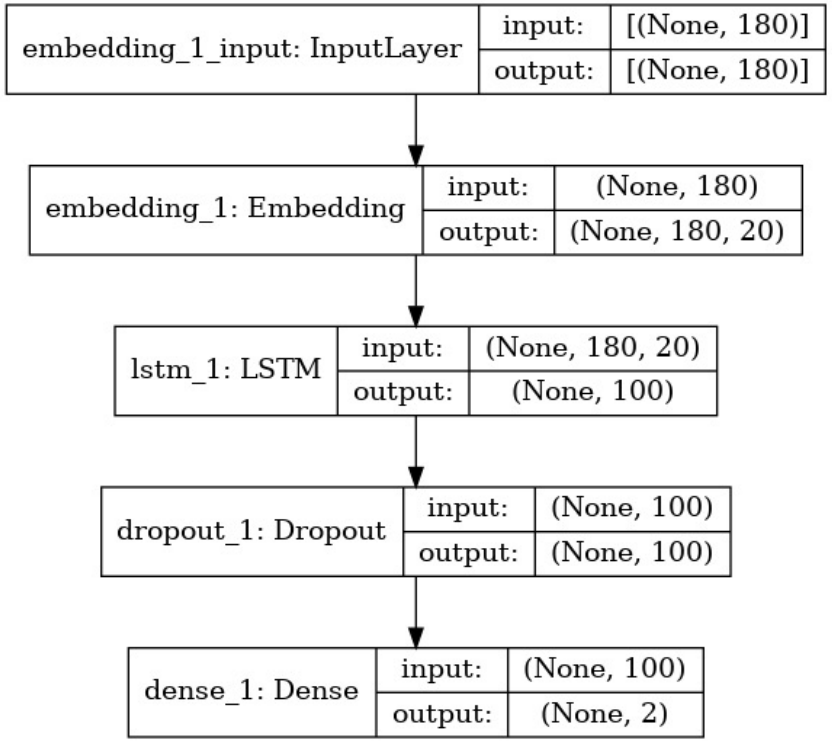


Performance



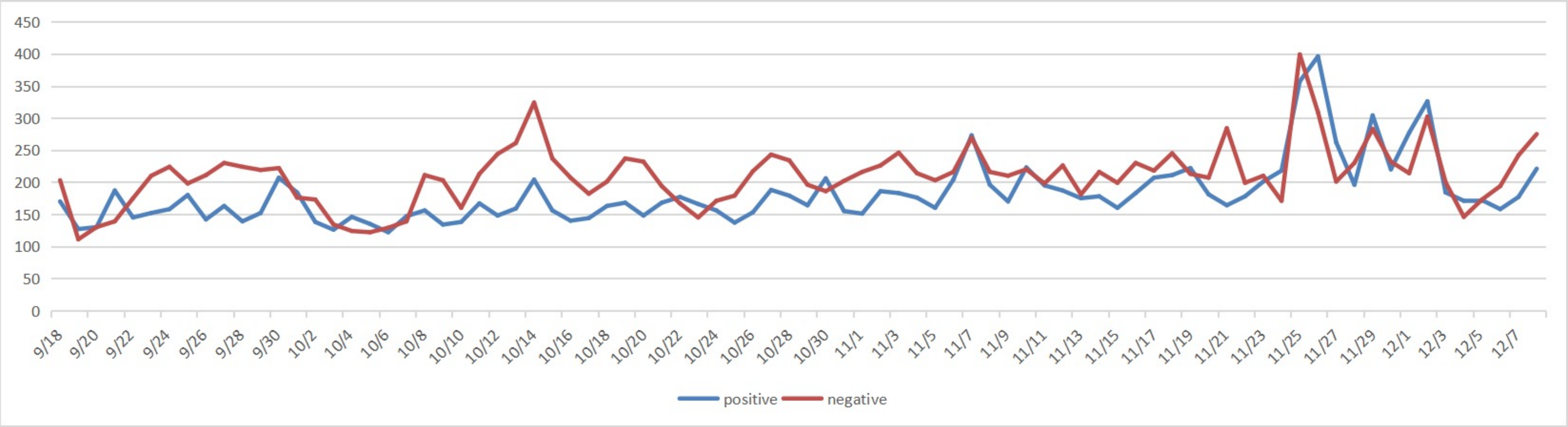
Performance——LSTM

```
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 180, 20)           86680
_________________________________________________________________
lstm_1 (LSTM)                (None, 100)               48400
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 202
=================================================================
Total params: 135,282
Trainable params: 135,282
Non-trainable params: 0
_________________________________________________________________
Epoch 1/5
2021-12-29 03:09:58.698097: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185]
ered 2)
273/273 [==============================] - 67s 228ms/step - loss: 0.6369 - accuracy: 0.6425
Epoch 2/5
273/273 [==============================] - 62s 228ms/step - loss: 0.5723 - accuracy: 0.7121
Epoch 3/5
273/273 [==============================] - 62s 227ms/step - loss: 0.5325 - accuracy: 0.7441
Epoch 4/5
273/273 [==============================] - 62s 226ms/step - loss: 0.4999 - accuracy: 0.7611
Epoch 5/5
273/273 [==============================] - 62s 227ms/step - loss: 0.4713 - accuracy: 0.7807
```

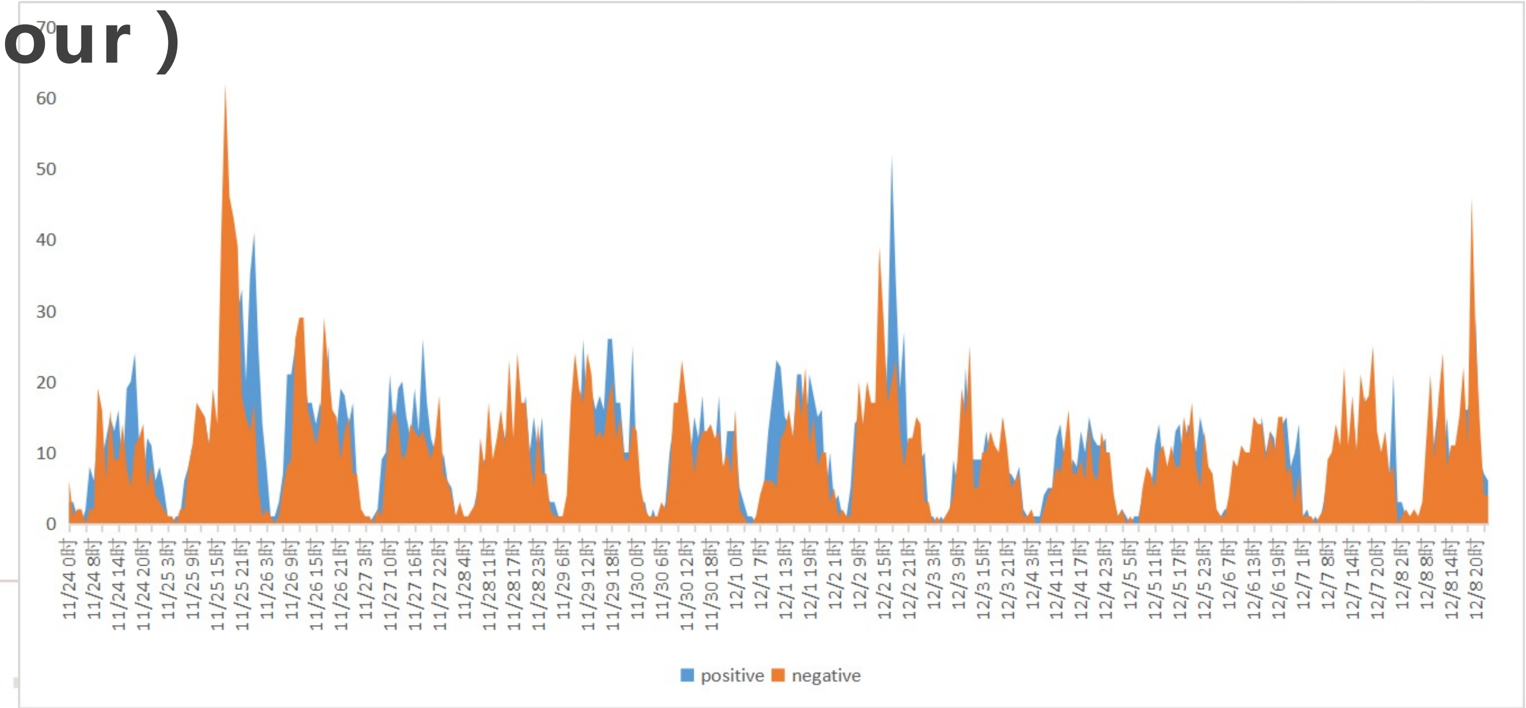


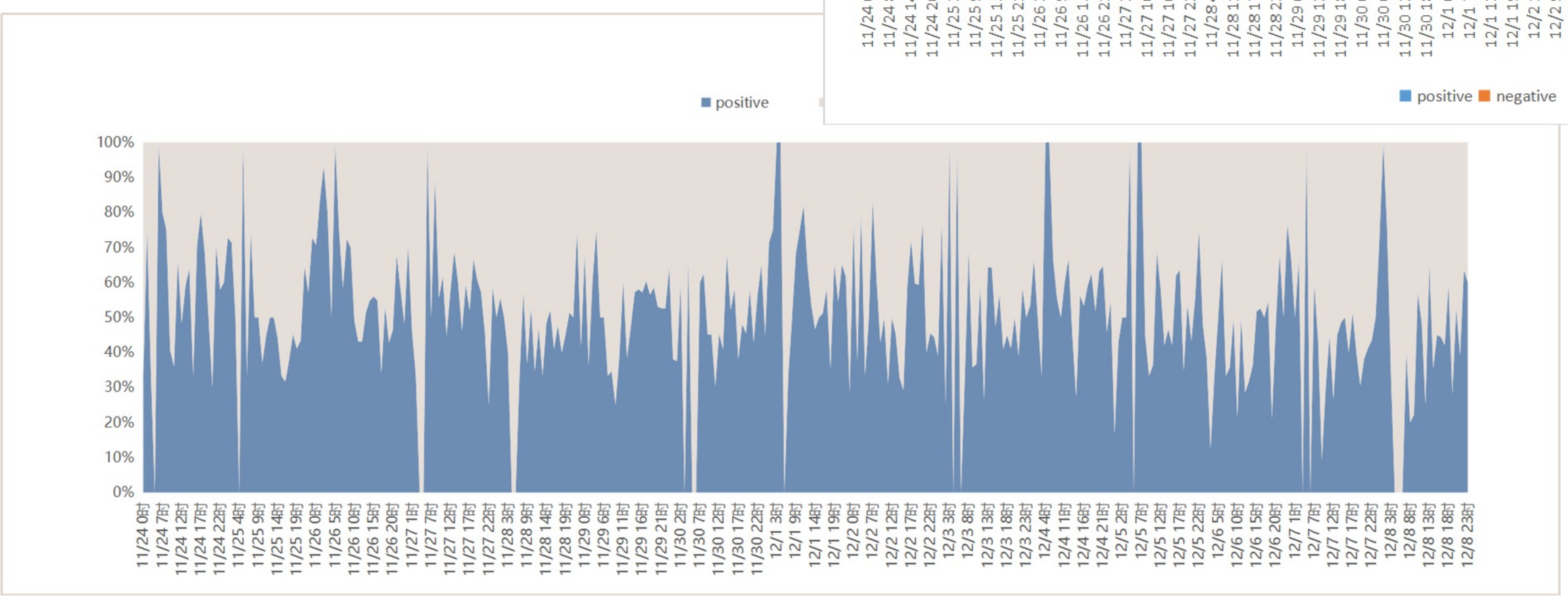

```
lication理解成APP，然后说开发APP应该考虑不同年龄的人兴趣不同阿巴阿巴[小丑]最绝的是我还自作聪明地说一个好的APP或许也应该让所有人受益，然后给到parterner一个argue的
点，他照着我的错误理解顺下去了，然后我们带着我的错误理解尬聊了三分钟[笑哭]最后考完我的parterner还对我诚挚地说了谢谢[麻了]希望他查完题目不会想打我[笑哭] 0 0
又是被安中充电桩气死的一天[愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒] 0 0
每次看自己小时候的照片都会很心动最爱的是眼睛感觉自己越长越残了[笑哭] 0 1
网课日常[狗头]上一次课间休息的时候，还听舍友跟我炫耀这椅子有多么舒服[笑哭][笑哭]。后来渐渐就没声了[笑哭][笑哭]，因为在听课也没怎么注意。刚刚一扭头发现她睡着了[可怜]
[可怜][笑哭] 0 1
汤圆老师的线代怎么还不出呀，宝宝想知道有没有挂科[滑稽][滑稽][滑稽] 0 1
自封校看网课以后已经三天没吃早饭了[狗头] 0 1
图片来自朵朵，如有侵权请联系删除 1 0
怎么会有这么肾虚的室友啊外面16度 他把空调开26度制热外面20度 他把空调开29度制热外面30度 他不让开空调笑死我了 0 0
请假回家了，现在是不是回不去学校了[笑哭][笑哭] 1 0
？这是真的嘛 0 0
请问下次更新什么时候?不要不识好歹谢谢[笑哭][笑哭] 1 1
堕落街缙云烧饼换人了耶，难道是因为上次十大被看见了（ 1 1
为什么电设实验报告这么麻烦，0.5学分，好多实验，每个实验报告都得3个小时写[伤心][伤心] 0 0
我tm心态崩了呀！[愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒][愤怒] 0 0
模型在测试集上的准确率为: 0.6542827657378741.
```

Sentiment change (day) 09/18-12/07



Sentiment change (hour) 11/24-12/08





Wordcloud Analysis

	cat	review	cat_id	clean_review	cut_review
0	1	宿舍终于收拾好啦，更不想起床了救命😫 😭	0	宿舍终于收拾好啦更不想起床了救命	宿舍 终于 收拾 好 更 不想 起床 救命
1	1	某英语外教线上课[哼]as an island nation一出来，整个人都不好了后面回答问...	0	某英语外教线上课哼asanislandnation一出来整个人都不好了后面回答问题的居然还有...	英语 外教 线 上课 asanislandnation 出来 整个 人 都 不好 后面 回答...
2	1	优质pku大学生😅	0	优质pku大学生	优质 pku 大学生
3	1	在朵朵放三个月饼，大家自取。[龇牙]	0	在朵朵放三个月饼大家自取龇牙	朵朵 放 三个 月饼 大家 自取 龇牙
4	1	[征友|♂征♀]帮po的室友征友基本信息：宁波人，00年xgg,身高178cm，体重75kg...	0	征友征帮po的室友征友基本信息宁波人00年xgg身高178cm体重75kg颜值个人感觉有点小...	征友 征帮 po 室友 征友 基本 信息 宁波人 00 年 xgg 身高 178cm 体重 ...

Wordcloud Analysis

the analysis of top ten hottest posts

1. sentiment words : 哭 笑 伤心 **滑稽** 可怜 喜欢
2. social network level words : 大家 同学 朋友 人 室友

sentiment_label	counts
1	317
0	302

☑ 619 data

☑ Many of the top ten hot posts involve **emotional issues**, including roommate relationships, boyfriend-friend relationships, personal emotional issues, etc.

☑ The three words with the most significant emotional inclinations when the students posted are: **crying, laughing and funny**.

Wordcloud Analysis

analysis result of posts labeled “negative”

analysis result of posts labeled “positive”

同学？

Many of posts involved conflicts between classmates: including the embarrassment feeling when socialize, the anger of being unplugged from the battery car charger, and the social fear in class speeches and so on.

好看 喜欢

征友 吃 希望

发现 亲亲

Wordcloud Analysis

the analysis of the posts liked by 100 people

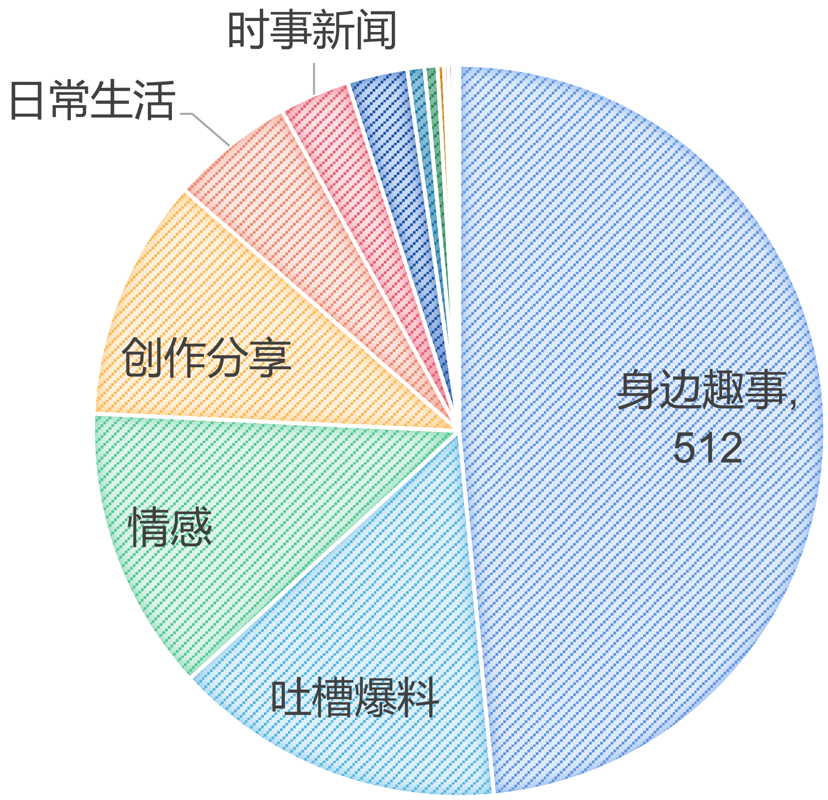
sentiment words：

哭 笑 伤心 **滑稽** **狗头** 希望 喜欢

degree words：

没觉得 可能 还好 不要 不 没 不是

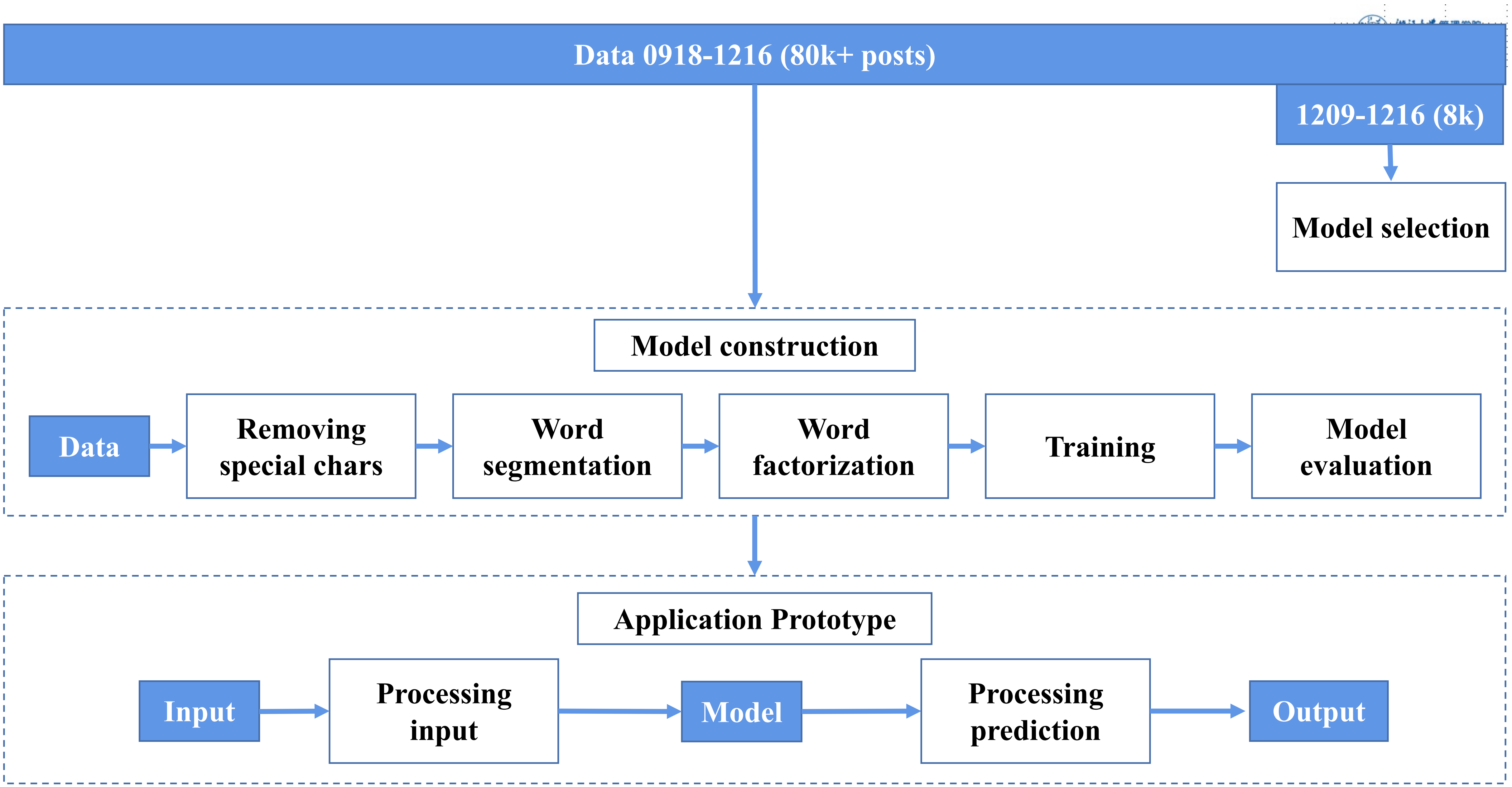
→主题：身边趣事 爆料吐槽 情感



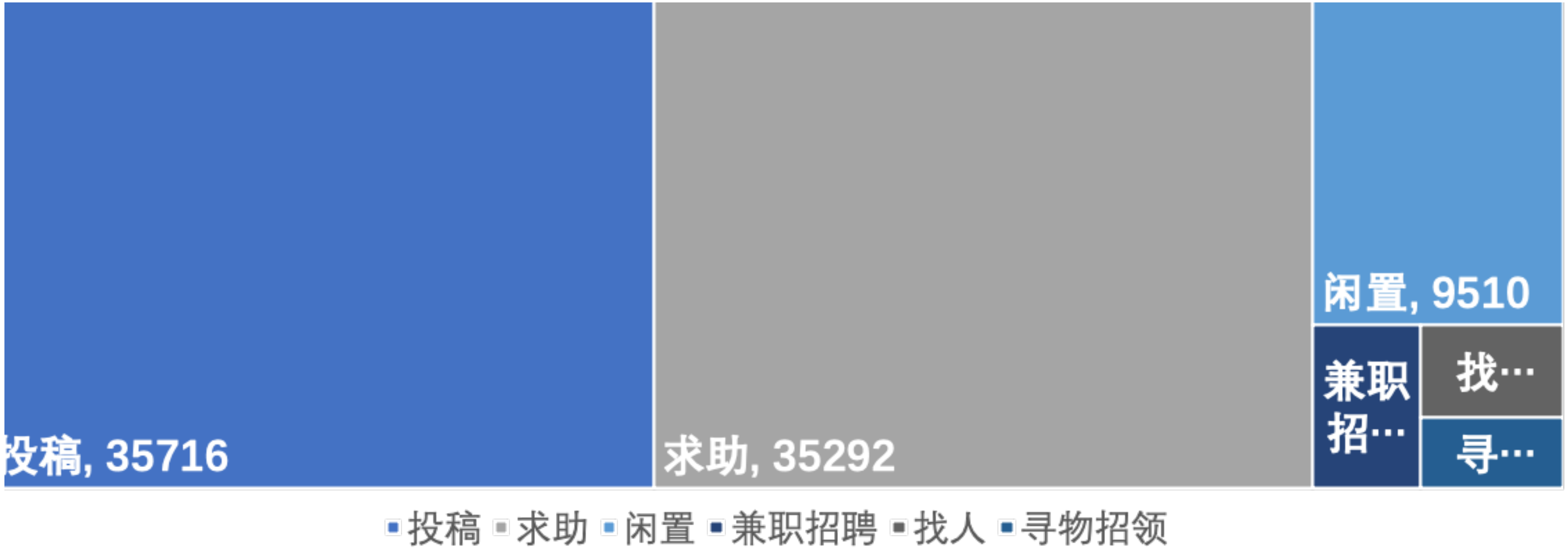
sentiment_label	count
1	573
0	481

Zheda Yunduoduo: Choose a topic before posting

What if: no need to choose?


Data 0918-1216 (80k+ posts)
1209-1216 (8k)
Model selection
Model construction
Data
Removing special chars
Word segmentation
Word factorization
Training
Model evaluation
Application Prototype
Input
Processing input
Model
Processing prediction
Output

Proportions of 6 Topics



Word Clouds for 6 Topics

投稿

闲置

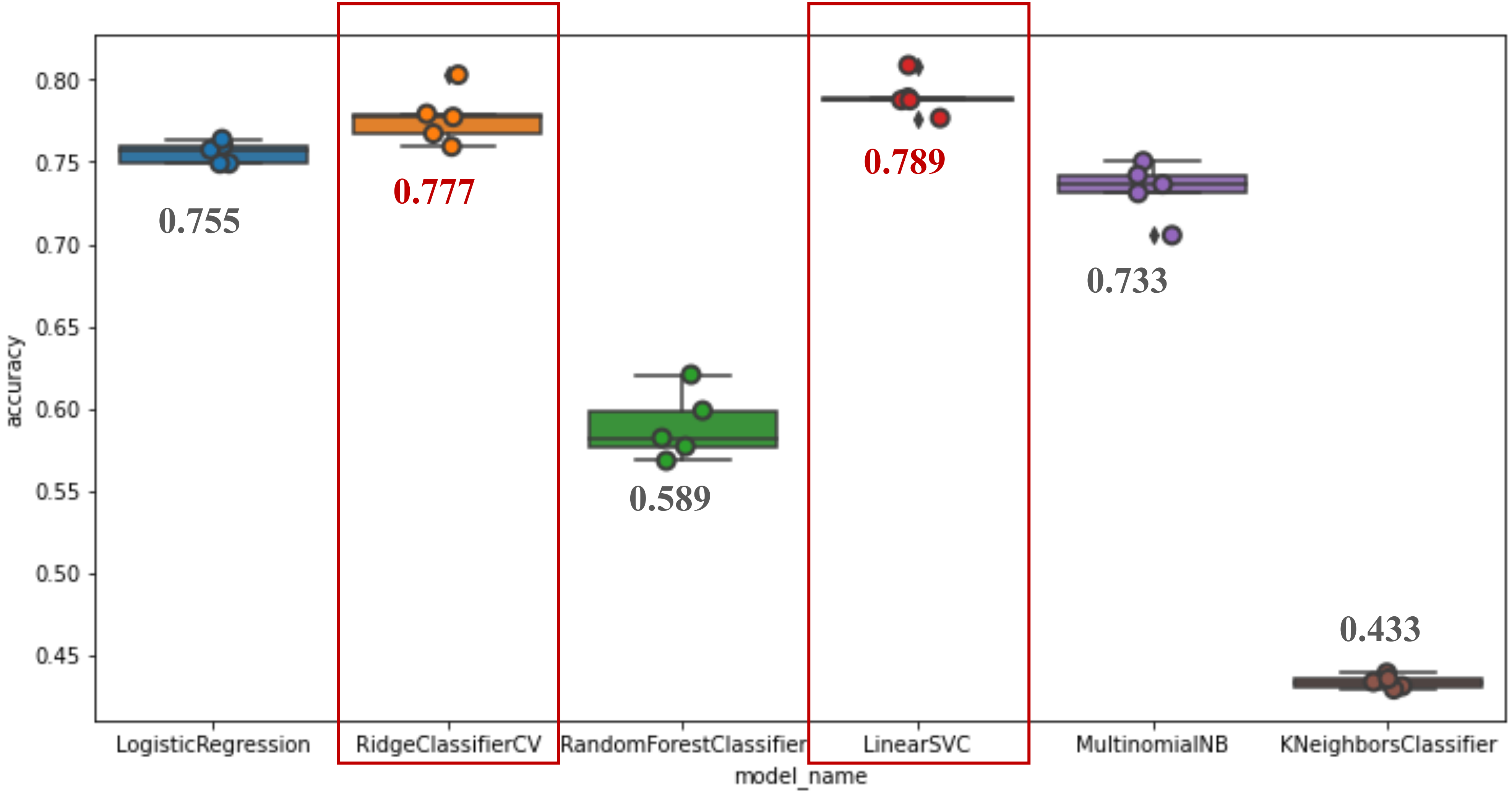
找人

求助

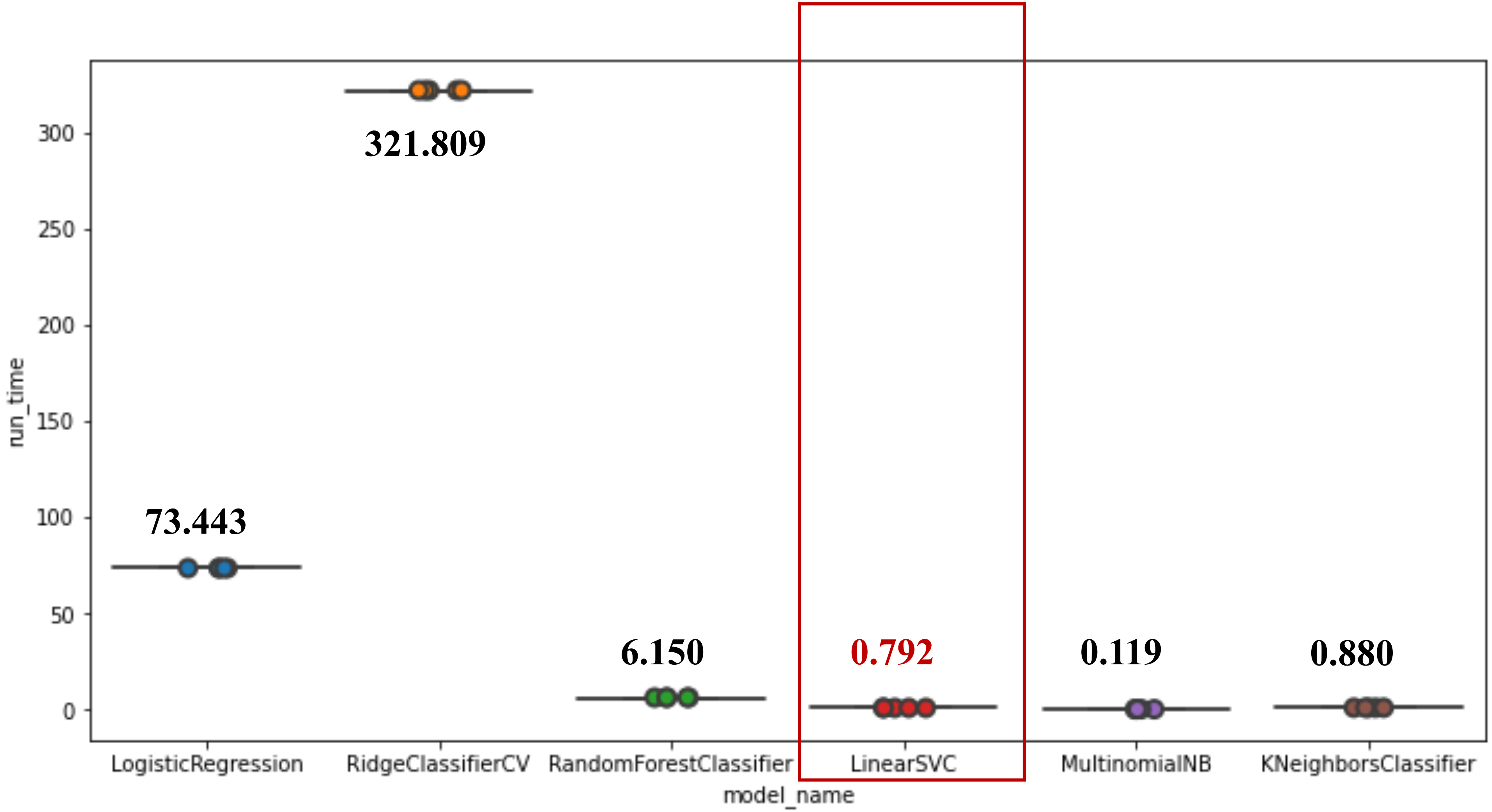
兼职/招聘

寻物/招领

Model Evaluation: Accuracy as Criterion



Model Evaluation: CPU Time as Criterion



Model Evaluation: Linear SVC

Modelled using 1-week data

Model Evaluation: Linear SVC (L2-penalized)

Modelled using 3-month data

	precision	recall	f1-score	support
投稿	0.79	0.90	0.84	10588
闲置	0.94	0.85	0.89	2853
找人	0.64	0.08	0.14	375
求助	0.85	0.78	0.81	986
兼职招聘	0.91	0.77	0.83	286
寻物招领	0.76	0.53	0.63	505
WAvg	**0.83**	**0.83**	**0.83**	25322

Application prototype

Built using *Gradio* and reposited on *Huggingface*

TEXT

张晓明同学，你的饭卡落在体育场了，我帮你放在入口处了，记得去拿

Clear | Submit

CATEGORY 0.00s

寻物/招领

Screenshot | Flag

TEXT

有没有人可以帮我从菜鸟驿站拿个包裹，有偿

Clear | Submit

CATEGORY 0.00s

求助

Screenshot | Flag

TEXT

马上要期末考了，好紧张

Clear | Submit

CATEGORY 0.00s

投稿

Screenshot | Flag

Thank you!

Text mining and classifications in an alumni forum

December, 2021