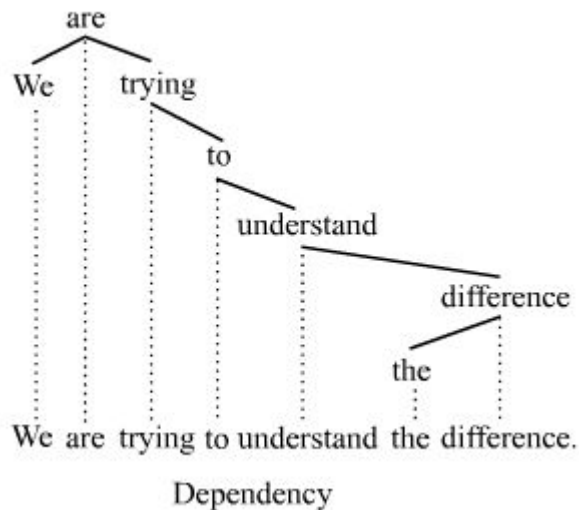
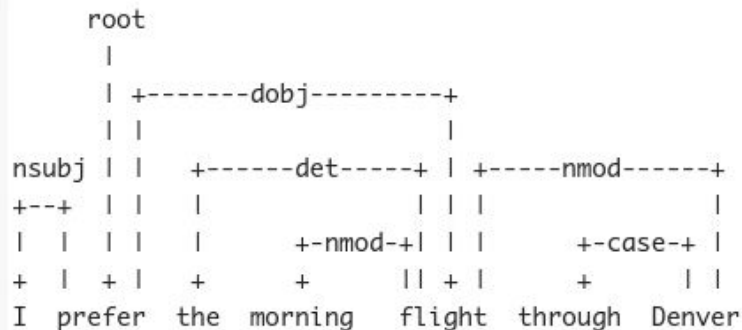

DLHLP HW4-2

謝濬丞 紀伯翰

Dependency Parsing Tree

1. Dependency parsing is the task of extracting a dependency parse of a sentence that represents its grammatical structure and defines the relationships between head words and words, which modify those heads.

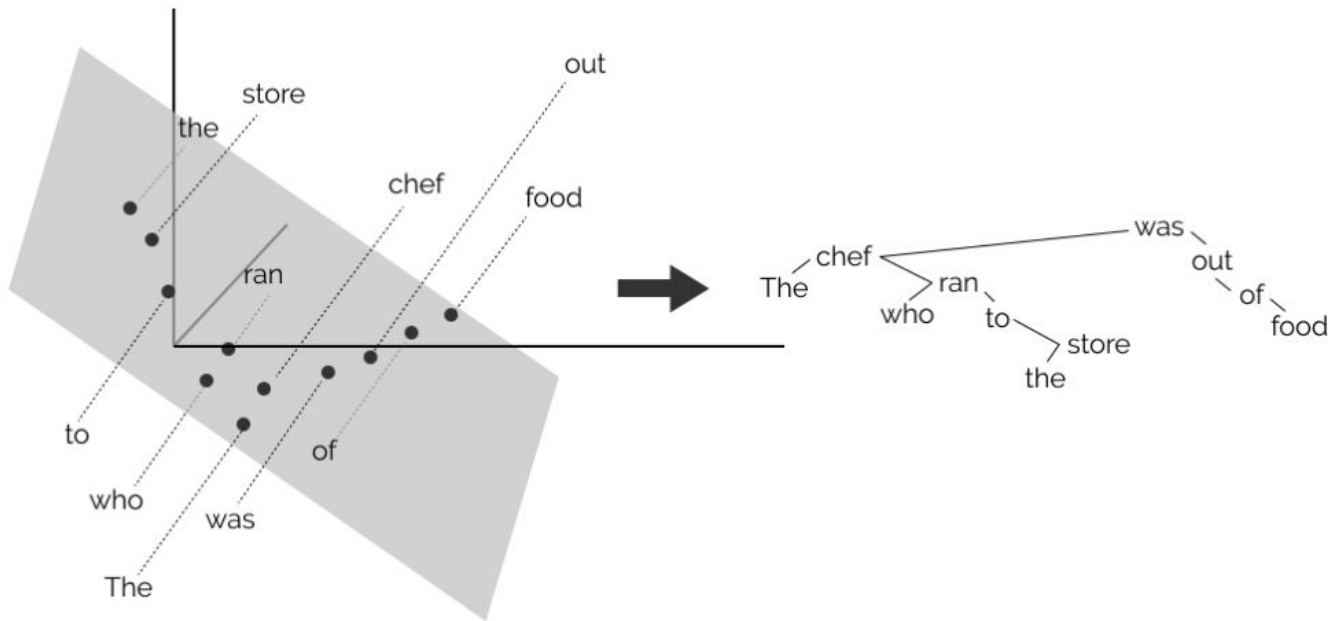


來源:

https://en.wikipedia.org/wiki/Dependency_grammar ,
http://nlpprogress.com/english/dependency_parsing.html

Tree structure of contextualized word embeddings learnt by Bert

The **chef** who ran to the store **was** out of food.



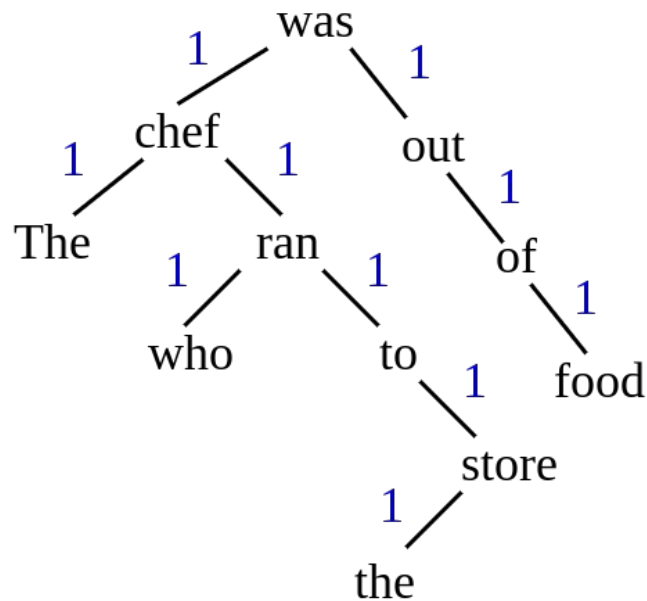
Definition 1

According to the tree, we define the distance between two neighbor is 1.

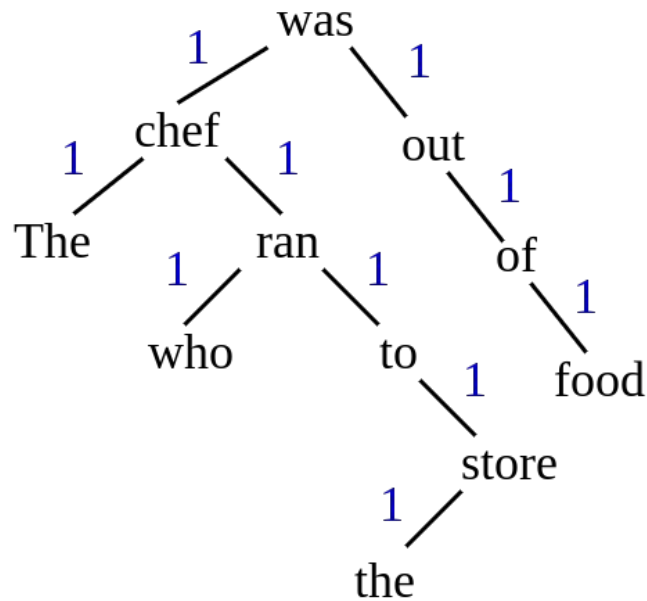
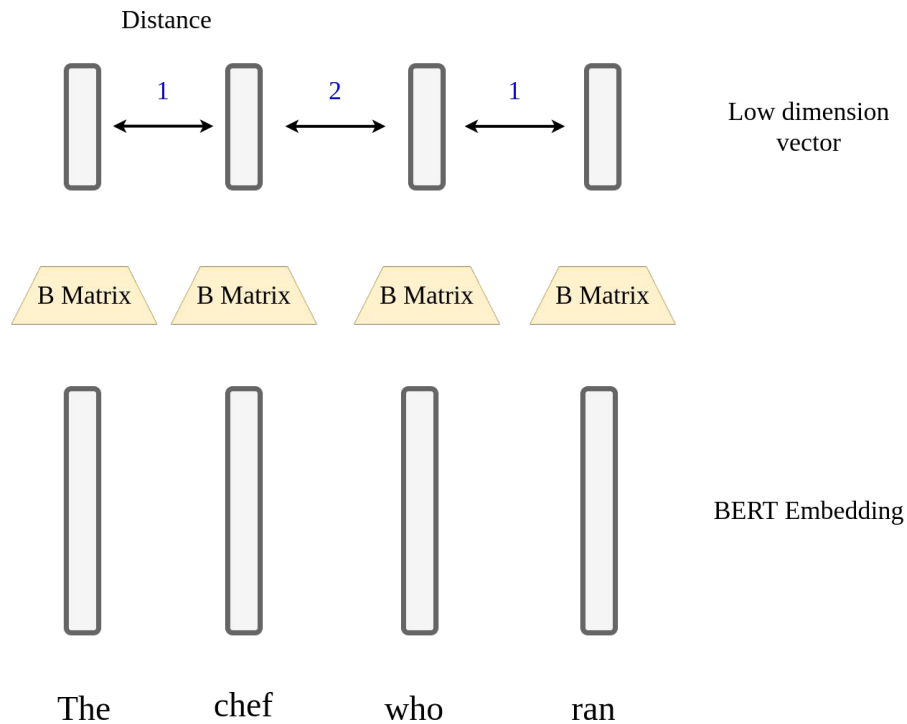
The operations are simple. First, subtract the two vectors after projection ,and then take the square of L2 norm of the subtracted vector, which is the square of their distance (for simplicity ,we simply use the square distance instead of the real one).

And use L1 loss to optimize the ground truth distance.

All parameters will be tuned is the projection matrix B. (No BERT model)



Process Detail



Definition 1

- Distance of high-dimensional vectors $d = (h_i - h_j)^T (h_i - h_j)$
- By definition, a valid distance metric is semi-positive definite
- We can learn a matrix B s.t $d_B(\mathbf{h}_i, \mathbf{h}_j) = (\mathbf{h}_i - \mathbf{h}_j)^T A (\mathbf{h}_i - \mathbf{h}_j) = [B(\mathbf{h}_i - \mathbf{h}_j)]^T B(\mathbf{h}_i - \mathbf{h}_j)$
with $A = B^T B$
- Which is equivalent to finding the subspace where the dependency tree is embedded.
- For a sentence of length N , with all $N(N-1)/2$ distances known, we can reconstruct an **undirected** dependency parse tree.

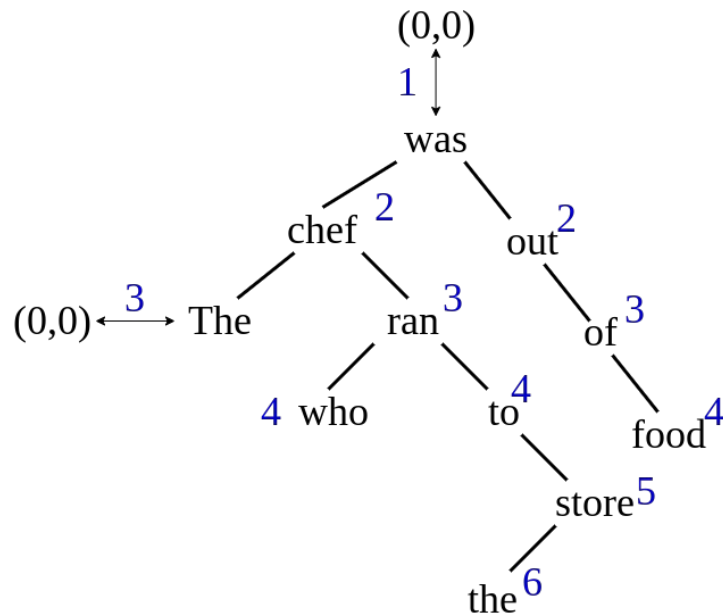
Definition 2

According to the tree, we define another distance like right picture.

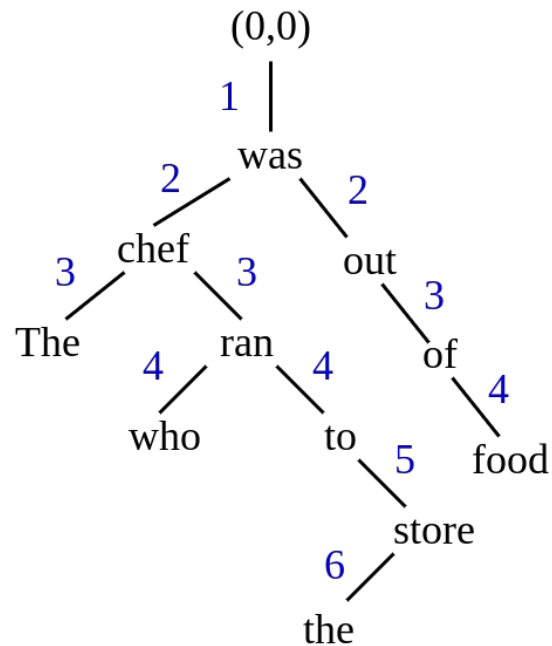
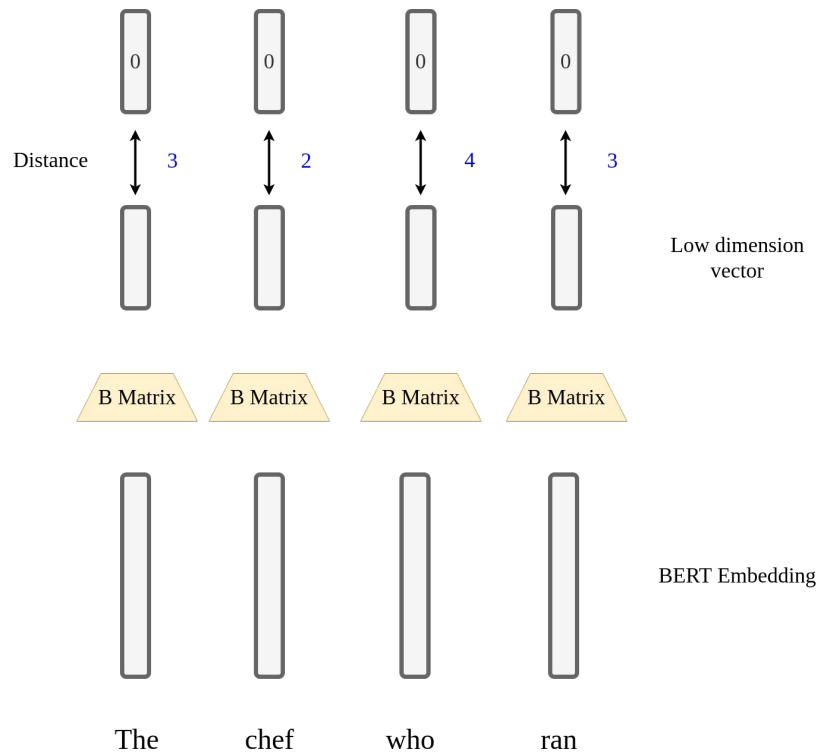
The operations are simple. All our procedure is just calculate the square of L2 norm (the distance) of the vector after projection.

And use L1 loss optimize the ground truth distance.

All parameters be tuned is the projection matrix B. (No BERT model)



Process Detail



Procedure

<https://github.com/Sologa/structural-probes>

Dataset (Chinese Treebank):

https://drive.google.com/open?id=1KMC-JYZ1wHYMFrAz_7iEiBWiGyBz94S4

(Copyrights belongs to LDC; Please do not distribute)

Bonus: Regression v.s. Rank loss

L1 loss forces the model to regress on the exact value of the true distances.

What if we simplify the task of getting the exact depth of the dependency tree right, to getting just the ordering of the depth right ?

Instead, minimize a pair-wise learning-to-rank loss (d: depth):

$$L_{\text{dist}}^{\text{rank}} = \sum_{i,j>i} [1 - \text{sign}(d_i - d_j)(\hat{d}_i - \hat{d}_j)]^+$$

You can also take the square out of the vector norm and see if regression still performs better.

Scoring: Submission

- Create a folder 'hw4-2' under your team Github repo
- 'hw4-2/' contains:
 - report.pdf
 - loss.py in structural probes **if you have completed the bonus**. And you should redo report problem 2 with rankloss.

[Report Template](#)

DLHLP HW4-3

陳延昊 吳宗翰

Attention

沒有強制要用Sample Code!

沒有強制要用Sample Code!

沒有強制要用Sample Code!

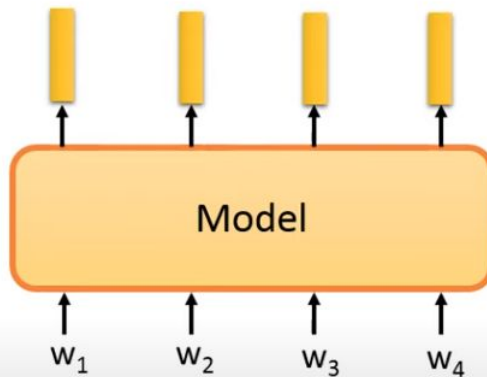
Question Answering

Context

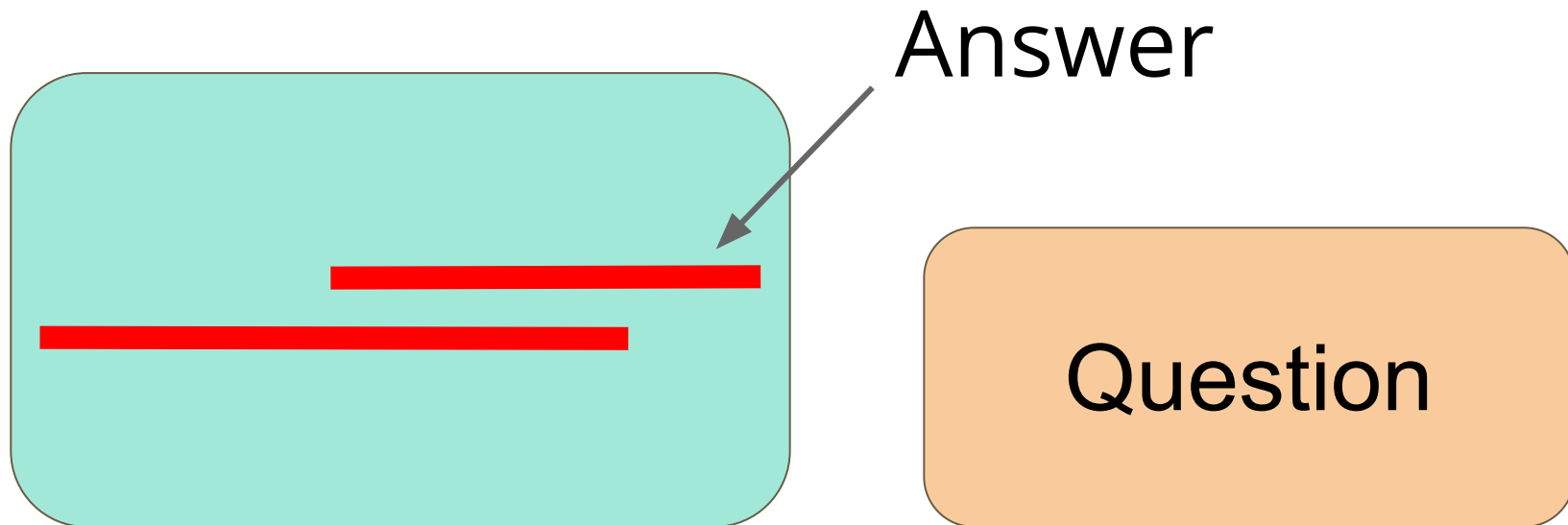
Question

Model

Masking Input



Question Answering



Code

https://github.com/YNNEKUW/DLHLP_HW4-3

Dataset

In Github

Kaggle rules

- Website: [DLHLP\(2020, Spring\) HW4-3](#)
- Your team name should be in [team_github_id]_[any_string]
e.g. daikin_大金
- 5 submission per team & per day
- Using any extra kaggle account to submit is cheating!
- Choose two submission as the final submission for private leaderboard before deadline
- Extra training data for fine-tuning is not allowed, but it is okay to use other pretrained models (please cite the source clearly in your report)

Kaggle Evaluation Metric

The evaluation metric for this competition is **Mean F1-Score**. The F1 score, commonly used in information retrieval, measures accuracy using the statistics precision p and recall r . Precision is the ratio of true positives (tp) to all predicted positives ($tp + fp$). Recall is the ratio of true positives to all actual positives ($tp + fn$). The F1 score is given by:

$$F1 = 2 \frac{p \cdot r}{p + r} \text{ where } p = \frac{tp}{tp + fp}, \quad r = \frac{tp}{tp + fn}$$

The F1 metric weights recall and precision equally, and a good retrieval algorithm will maximize both precision and recall simultaneously. Thus, moderately good performance on both will be favored over extremely good performance on one and poor performance on the other.

Baselines (5%)

Public Simple Baseline: 0.75544 (2 pt)

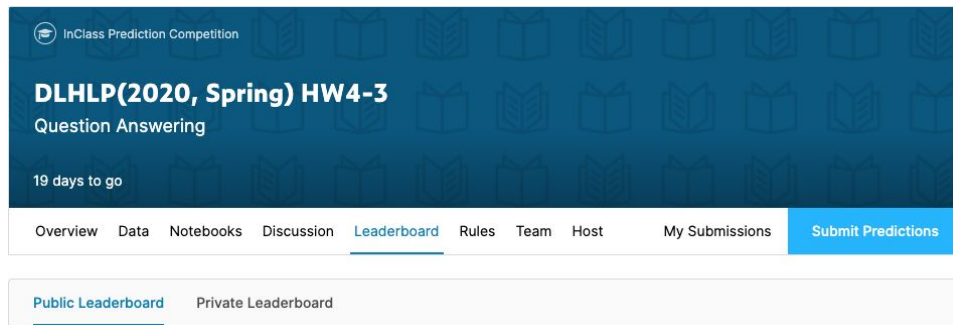
Public Strong Baseline: 0.75945 (1 pt)

Private Simple Baseline: (?) (1 pt)

Private Strong Baseline: (?) (1 pt)

P.S. private score will be shown
after the deadline

Both would be easy to beat :p



The screenshot shows the interface for the "DLHLP(2020, Spring) HW4-3 Question Answering" competition. At the top, it says "InClass Prediction Competition" with a clock icon. Below that, the competition title "DLHLP(2020, Spring) HW4-3" and "Question Answering" are displayed. A timer indicates "19 days to go". A navigation bar contains links: Overview, Data, Notebooks, Discussion, Leaderboard (which is underlined), Rules, Team, Host, My Submissions, and a blue "Submit Predictions" button. At the bottom, there are two tabs: "Public Leaderboard" (which is underlined) and "Private Leaderboard".

Report Questions (5%)

1. Please give some examples predicted correctly and incorrectly respectively. At least one for each case is required. Screenshot recommended. (2 points)
2. Which hyper hyperparameter(s) should be modified in order to reach better performance? Please explain why it works briefly. (e.g. learning_rate, batch_size, warmup_steps, layer_norm_eps, attention_probs_dropout_prob) (3 points)

[Template](#)

Bonus (2%)

- Presentation in class
- Selection criteria
 1. Innovation
 2. Different models compare bert-base-chinese
 - Existing github is valid, but you have to understand, explain, and cite it.
 3. Good performance
- The team quota and the presentation time will be announced based on the time we have.

Scoring: Submission

- Create a folder 'hw4-3' under your team Github repo
- 'hw4-3/' should contains:
 - report.pdf
 - download.sh
 - reproduce.sh
 - **config.json (Optional. If you use the sample code, uploading this is recommended.)**
 - other files and directories
- We restrict Python version 3.6.8 and must compatible with [these package](#)
- Scoring
 - Report (5%)
 - Kaggle (5%) (over baseline + successfully reproduce)

Reproduce

- We will run the following command to reproduce:

```
```bash download.sh <OUTPUT_MODEL_PATH>```
```

```
```bash reproduce.sh <OUTPUT_MODEL_PATH> <TEST_DATA_PATH> <OUTPUT_FILE_PATH>
```

e.g.

```
```bash download.sh ./pytorch_model.bin```
```

```
```bash reproduce.sh ./pytorch_model.bin ./hw4-3_test.json ./output.csv```
```

Github maximum capacity

- Use Dropbox to put your model, use 'wget' to download
- Dropbox Tutorial:
https://docs.google.com/presentation/d/1SsleIij9ZOEN_TGdbAS1oWcl6bT1uSTI6b5_u2wdDc/edit?usp=sharing
- Your shell script files should be able to download the model automatically

Code (for bonus)

Transformer:

[huggingface/transformers: 🤖 Transformers: State-of-the-art Natural Language Processing for Pytorch and TensorFlow 2.0.](#)

Models:

<https://github.com/ymcui/Chinese-BERT-wwm?fbclid=IwAR13Cr8iQEIsFTU60jYhJPS9yixjU817hMrE59h9tdFdRVefgy8d9SuuRX0>

Deadline

2020.5.27 9:00

Late submission policy

- You can submit file until 3 days after deadline
- The score will be calculated as:

$$\text{score}_{\text{final}}(\text{hr}) = \begin{cases} \text{score}_{\text{original}} \times 0.985^{\text{hr}} & , \text{hr} \leq 72 \\ 0 & , \text{hr} > 72 \end{cases}$$

- Late submission form would be announced after deadline

Attention

沒有強制要用Sample Code!

沒有強制要用Sample Code!

沒有強制要用Sample Code!



FB Group:
Deep Learning for Human
Language Processing
(2020, Spring)

Q&A

dlhlp.ta@gmail.com