NLP data representation

Lab 2

Natural Language Process (AI5304)

Spring 2022

GIST Intelligence Representation &Reasoning Lab

Outline

- Preliminary
  - Tokenization
  - Lemmatization
  - Character Embedding
- Task – Sentence Classification
  - Step 1. Tokenize the input Sentence
  - Step 2. Lemmatize the tokenized words
  - Step 3. Word Representation using Char Embedding
  - Step 4. Train your sentence classification model
  - Step 5. Evaluate the performance of your trained model on test set

Preliminary

- Tokenization

Tokenizers divide a corpus into lists of tokens

The unit of token varies depending on the corpus, but the token is usually defined in a meaningful unit

Usually, tokenization means dividing a sentence to the list of words

e.g.

```
input = 'I had beautiful flowers'.lower()
print("Input:", input)
print("Output:", tokenization(input))
```

```
Input: i had beautiful flowers
Output: ['i', 'had', 'beautiful', 'flowers']
```

Corpus	Token
Document	Sentence
Sentence	Word
⋮	⋮

Preliminary

- Lemmatization

The process of grouping inflected forms together as a single base form

*vs. stemming**
: lemmatization finds more accurate stem than stemming, because lemmatization considers the syntactic and semantic meaning of word such as grammar, PoS tag**, and so on.

Usually use the external information such as WordNet

e.g.

```
input = 'I had beautiful flowers'.lower()
print("Input         :", input)
print("Output [Lemma]:", lemmatization(input))
print("Output [Stem ]:", stemming(input))
```

```
Input         : i had beautiful flowers
Output [Lemma]: ['i', 'have', 'beautiful', 'flower']
Output [Stem ]: ['i', 'had', 'beautiful', 'flow']
```

* Stemming
  : Extract “stem” from word
  : Simply cut the ending of a word

** Part of Speech (PoS) tagging
: the process of marking up the word in a sentence as corresponding to part-of-speech, which is a category of words, such as noun, verb, adverb, and so on

I	like	to	read	books
PRP	VBP	TO	VB	NNS

Preliminary

- Lemmatization

  * WordNet

PRINCETON UNIVERSITY

WordNet

A Lexical Database for English

- What is WordNet
- People
- News
- Use Wordnet Online
- Download
- Citing WordNet
- License and Commercial Use
- Related Projects
- Documentation
- Publications
- Frequently Asked Questions

What is WordNet?

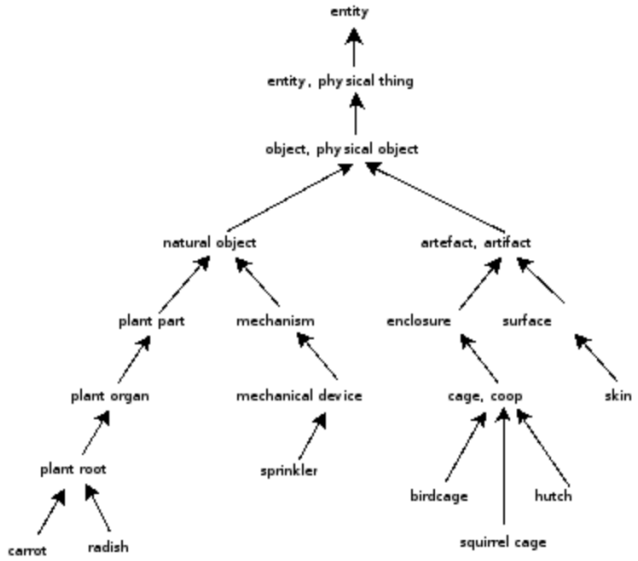
Any opinions, findings, and conclusions or recommendations expressed in this material are those of the creators of WordNet and do not necessarily reflect the views of any funding agency or Princeton University.

When writing a paper or producing a software application, tool, or interface based on WordNet, it is necessary to properly cite the source. Citation figures are critical to WordNet funding.

About WordNet

WordNet® is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. The resulting network of meaningfully related words and concepts can be navigated with the browser. WordNet is also freely and publicly available for download. WordNet's structure makes it a useful tool for computational linguistics and natural language processing.

WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings. However, there are some important distinctions. First, WordNet interlinks not just word forms—strings of letters—but specific senses of words. As a result, words that are found in close proximity to one another in the network are semantically disambiguated. Second, WordNet labels the semantic relations among words, whereas the groupings of words in a thesaurus does not follow any explicit pattern other than meaning similarity.



https://wordnet.princeton.edu/
https://www.cs.princeton.edu/courses/archive/spr07/cos226/assignments/wordnet.html

Preliminary

- (remind) Word Embedding

word dictionary

word	index
<pad>	0
<unk>	1
have	2
beautiful	3
⋮	⋮
flower	2599

one-hot representation

word	vector $(v \in R^{2600})$
<pad>	[1, 0, 0, 0, 0, …, 0]
<unk>	[0, 1, 0, 0, 0, …, 0]
have	[0, 0, 1, 0, 0, …, 0]
beautiful	[0, 0, 0, 1, 0, …, 0]
⋮	⋮
flower	[0, 0, 0, 0, 0, …, 1]

How to make it learnable?

dense representation

word	vector $(v \in R^{N})$
<pad>	[0.2, 0.1, -0.6, …, 0.7]
<unk>	[-0.3, 0.7, 0.4, …, -0.2]
have	[0.9, 0.8, 0.6, …, 0.1]
beautiful	[-0.4, -0.3, -0.6, …, 0.5]
⋮	⋮
flower	[0.8, 0.5, 0.6, …, -0.3]

Preliminary

- Character Embedding

character dictionary

char	index
<P>	0
<U>	1
a	2
b	3
⋮	⋮
?	53

one-hot representation

char	vector $(v \in R^{54})$
<P>	[1, 0, 0, 0, 0, …, 0]
<U>	[0, 1, 0, 0, 0, …, 0]
a	[0, 0, 1, 0, 0, …, 0]
b	[0, 0, 0, 1, 0, …, 0]
⋮	⋮
?	[0, 0, 0, 0, 0, …, 1]

dense representation

char	vector $(v \in R^{N})$
<P>	[0.2, 0.1, -0.6, …, 0.7]
<U>	[-0.3, 0.7, 0.4, …, -0.2]
a	[0.9, 0.8, 0.6, …, 0.1]
b	[-0.4, -0.3, -0.6, …, 0.5]
⋮	⋮
?	[0.8, 0.5, 0.6, …, -0.3]

How to get the dense representation about characters?

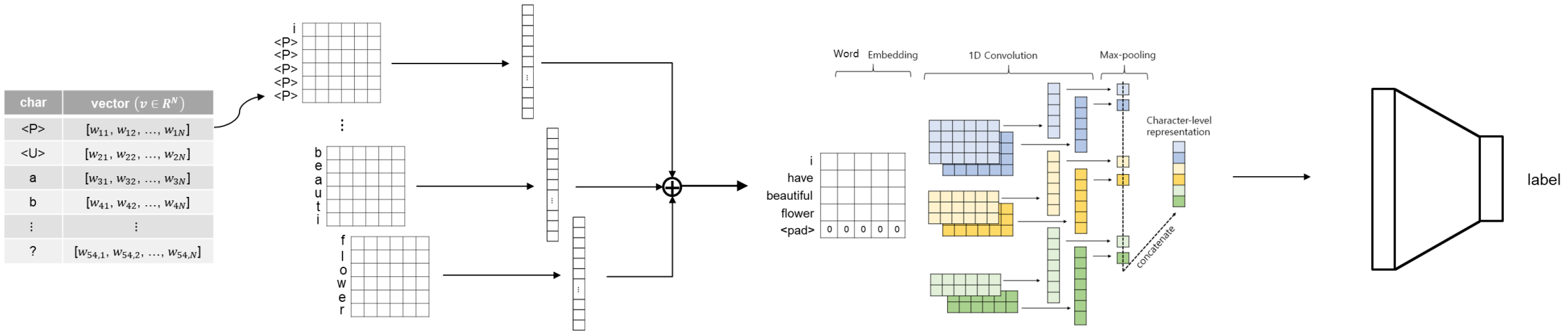
How to represent a sentence to a vector by using Character Embedding?

Task – Sentence Classification

- Overview

(Goal) Given a sentence, find the label of the sentence

1. Character embedding
2. word2vec from character embedding
3. sent2vec
4. Train Classifier



https://wikidocs.net/116193

Task – Sentence Classification

- Data Description

Input	sentence (lower case)
Output	label (# classes: 6)
# train data	4,500
# test data	500

```
pd.read_csv('data/sent_class.train.csv')
```

	sentence	label
0	? december scorpion firewall is journalism 197...	5
1	? station son spumante the wine what	1
2	? firewall srpska pointsettia find what	1
3	? seasons virginia shores whip broken flag what	5
4	is bible burma the lobster ? what	1
...	...	...
4495	? collection prostitute animal co-starred is h...	1
4496	? starring enzymes marketed directorial most r...	1
4497	? brothers 1978 is dc 0 washington o o fatalis...	3
4498	earth claws televised impossible another is de...	4
4499	? the has rules word costner kevin into come l...	1

4500 rows × 2 columns

```
pd.read_csv('data/sent_class.test.csv')
```

	sentence	label
0	was who steven randy glass most represented what	0
1	? the kimpo what	0
2	? faces amphibians 0 clothes www.questions.com...	0
3	earth you workers replies born workers slows n...	0
4	? tracy its yankees doodle pitch families viii...	0
...	...	...
495	was dog is milliseconds travels missionary gir...	0
496	? dentist you is twelve competitor written what	0
497	? upstaged sister-in-law quart calls humor mot...	0
498	? the there legend urban what	0
499	language the match what	0

500 rows × 2 columns

Task – Sentence Classification

- Step 1. Tokenize the input sentence use `word_tokenize` in NLTK

  Convert all sentences into the list of words by using tokenization

```
input = 'I had beautiful flowers'.lower()
print("Input:", input)
print("Output:", tokenization(input))
```

```
Input: i had beautiful flowers
Output: ['i', 'had', 'beautiful', 'flowers']
```

- Step 2. Lemmatize the tokenized words use `WordNetLemmatizer` in NLTK

  Lemmatize all words in the sentence

```
input = 'I had beautiful flowers'.lower()
print("Input:", input)
print("Output:", lemmatization(input))
```

```
Input: i had beautiful flowers
Output: ['i', 'have', 'beautiful', 'flower']
```

You should set the parameter `pos` to the proper tag for every word

e.g., `word = 'apple'`
`w_lemma = LEMMAFUNC(word, pos='n')`

Task – Sentence Classification

- Step 3. Word Representation using Char Embedding

character dictionary

char	index
<P>	0
<U>	1
a	2
b	3
⋮	⋮
?	53

dense representation

char	vector $(v \in R^N)$
<P>	[0.2, 0.1, -0.6, …, 0.7]
<U>	[-0.3, 0.7, 0.4, …, -0.2]
a	[0.9, 0.8, 0.6, …, 0.1]
b	[-0.4, -0.3, -0.6, …, 0.5]
⋮	⋮
?	[0.8, 0.5, 0.6, …, -0.3]

Make it Learnable!

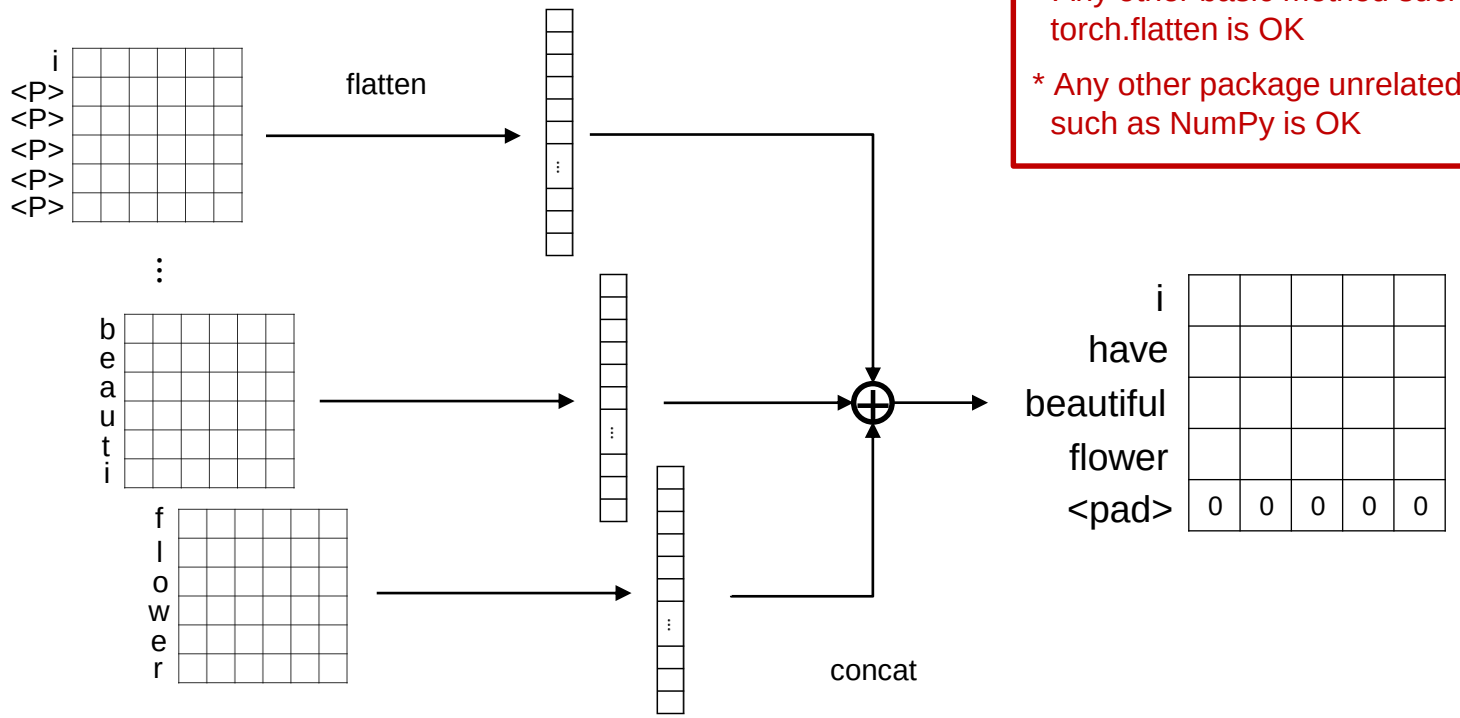
char	vector $(v \in R^N)$
<P>	$[w_{11}, w_{12}, \ldots, w_{1N}]$
<U>	$[w_{21}, w_{22}, \ldots, w_{2N}]$
a	$[w_{31}, w_{32}, \ldots, w_{3N}]$
b	$[w_{41}, w_{42}, \ldots, w_{4N}]$
⋮	⋮
?	$[w_{54,1}, w_{54,2}, \ldots, w_{54,N}]$

Task – Sentence Classification

- Step 3. Word Representation using Char Embedding

Convert word to vector from character embedding

char	vector $(v \in R^N)$
<P>	$[w_{11}, w_{12}, \ldots, w_{1N}]$
<U>	$[w_{21}, w_{22}, \ldots, w_{2N}]$
a	$[w_{31}, w_{32}, \ldots, w_{3N}]$
b	$[w_{41}, w_{42}, \ldots, w_{4N}]$
⋮	⋮
?	$[w_{54,1}, w_{54,2}, \ldots, w_{54,N}]$



* USE ONLY these modules in this step

  torch.nn.Module
  torch.nn.Parameter
  torch.nn.init_kaiming_uniform_

* Any other basic method such as torch.cat, torch.flatten is OK

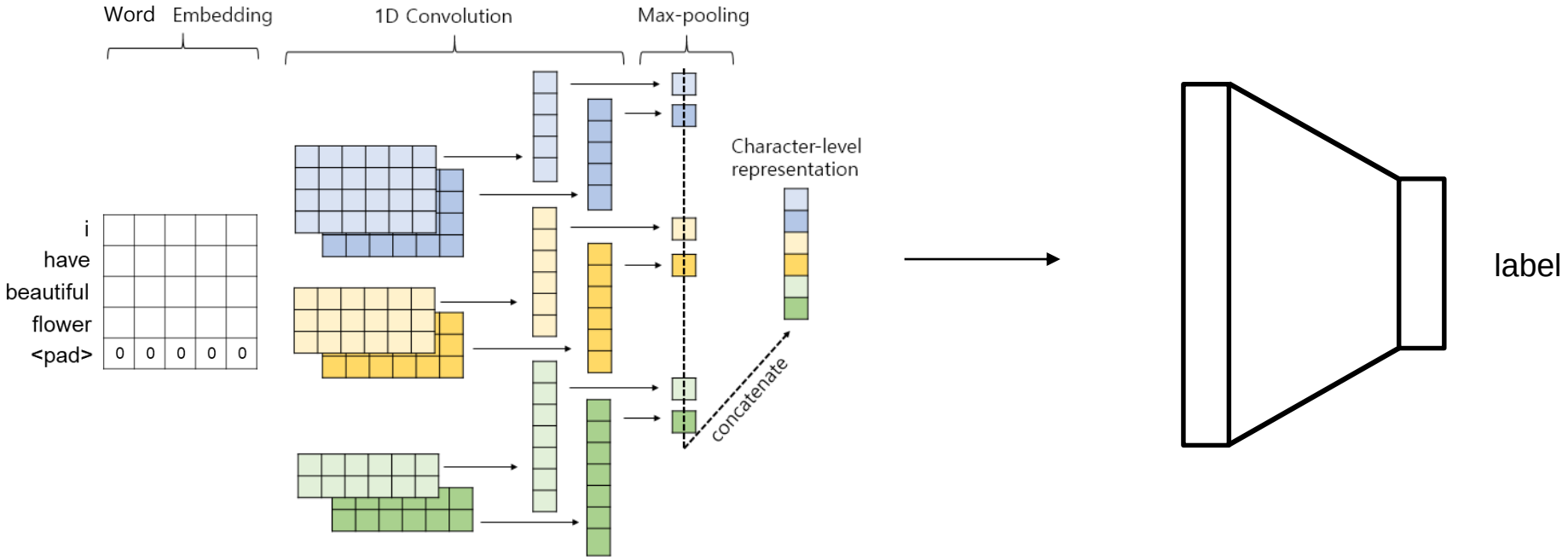
* Any other package unrelated to embedding such as NumPy is OK

Task – Sentence Classification

- Step 4. Train your sentence classification model

1. Convert sentence to vector from character embedding by using 1D CNN

2. Train the classifier



vec('i have beautiful flowers') = [0.9 0.7 -1.0 0.3 0.2 -0.5]

Task – Sentence Classification

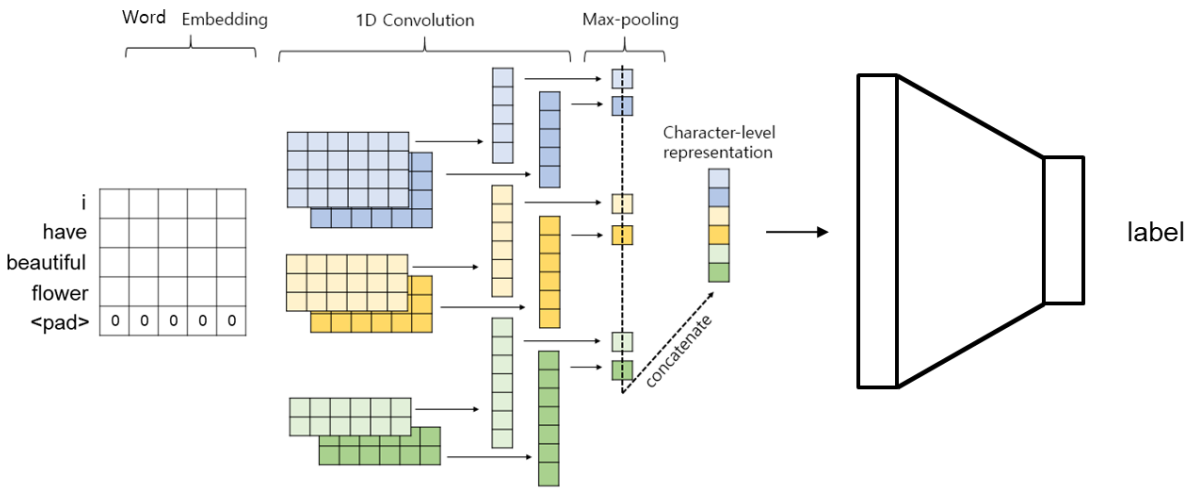
- Step 4. Train your sentence classification model

N_s: max length of sentence

N_w: max length of word

D_c: dimension of character vector

* Architecture

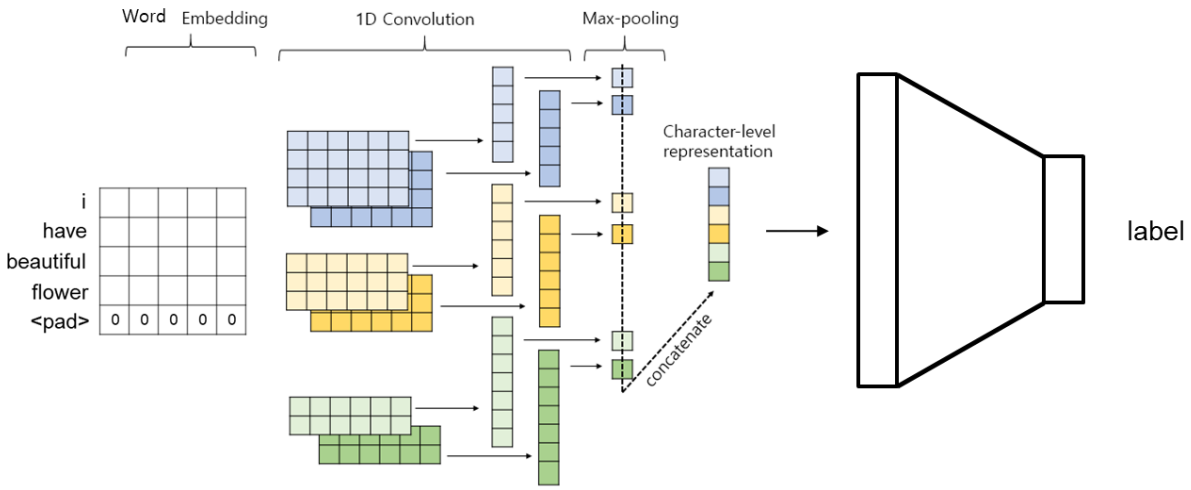


Input $\in \mathbb{R}^{N_s \times (N_w \times D_c)}$		
Conv (ch_{in}, ch_{out}) = (1,100) k: (2, $N_w \times D_c$) s: 1	Conv (ch_{in}, ch_{out}) = (1,100) k: (3, $N_w \times D_c$) s: 1	Conv (ch_{in}, ch_{out}) = (1,100) k: (4, $N_w \times D_c$) s: 1
batch normalization	batch normalization	batch normalization
ReLU	ReLU	ReLU
maxpool	maxpool	maxpool
concatenate		
FC - 100		
ReLU		
FC - # classes		
softmax		

Task – Sentence Classification

- Step 4. Train your sentence classification model

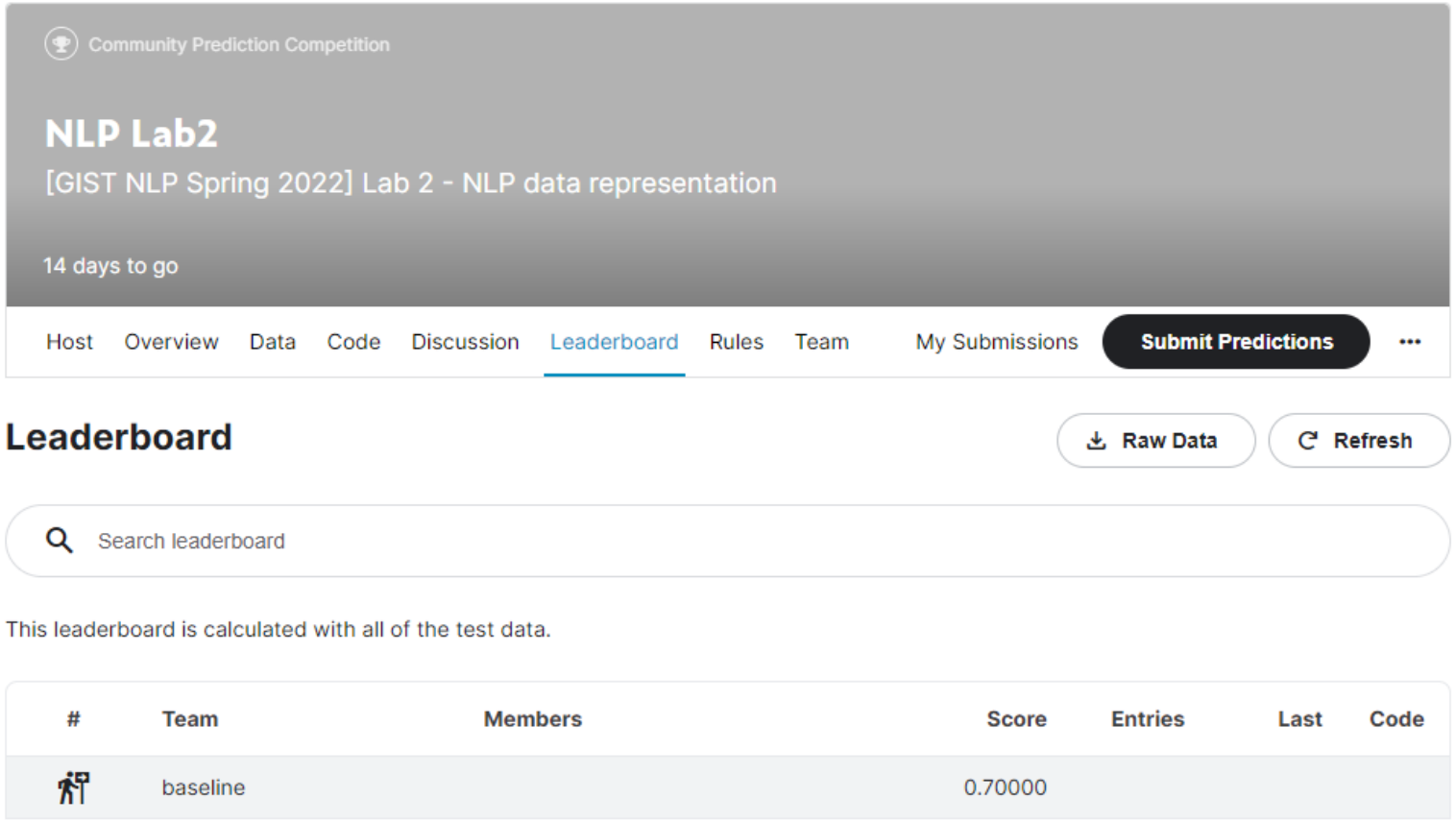
* Recommended hyper-parameter settings



$N_s \times (N_w \times D_c)$	20 x (10 x 100)
learning rate	0.001
# epochs	20
batch size	32
loss function	cross entropy
optimizer	adam

Task – Sentence Classification

- Step 5. Evaluate the performance of your trained model on test set



TO DO

- Task – Sentence Classification
  - Step 1. Tokenize the input sentence (use nltk.tokenize.word_tokenize) [1pt]
  - Step 2. Lemmatize the tokenized words (use nltk.stem.WordNetLemmatizer) [1pt]
  - Step 3. Word Representation using Character Embedding [5pts]
  - Step 4. Train your sentence classification model [3pts]
  - Step 5. Evaluate the performance of your trained model on test set [x1.0 or x0.5]

e.g., you got 8pts in Step1~4, and your model didn't show the performance equal to or better than baseline
-> (final score

Supplements

- NLTK package

  https://www.nltk.org/

- Character Embedding using 1D CNN

  https://wikidocs.net/116193

- Convolutional Neural Network for Sentence Classification

  https://aclanthology.org/D14-1181.pdf

- Embedding
  - cs224n – Lec01. Introduction and Word Vectors

    https://www.youtube.com/watch?v=8rXD5-xhemo

  - cs224n – Lec12. Subword Models

    https://www.youtube.com/watch?v=9oTHFx0Gg3Q

- CNN
  - cs224n – NLP with Deep Learning

    https://www.youtube.com/watch?v=EAJoRA0KX7I

Other Information

- Schedule
  - To be uploaded: 03.31(Mon) 11:00
  - Deadline: 04.13(Sun) 23:59
- Links
  - Kaggle Private Competition
  - Github classroom
- TA info
  - Jaehyoung Jeong (jaehyoung98@gm.gist.ac.kr)
- Module and Function List
  - We allow ONLY **PyTorch** in this assignment