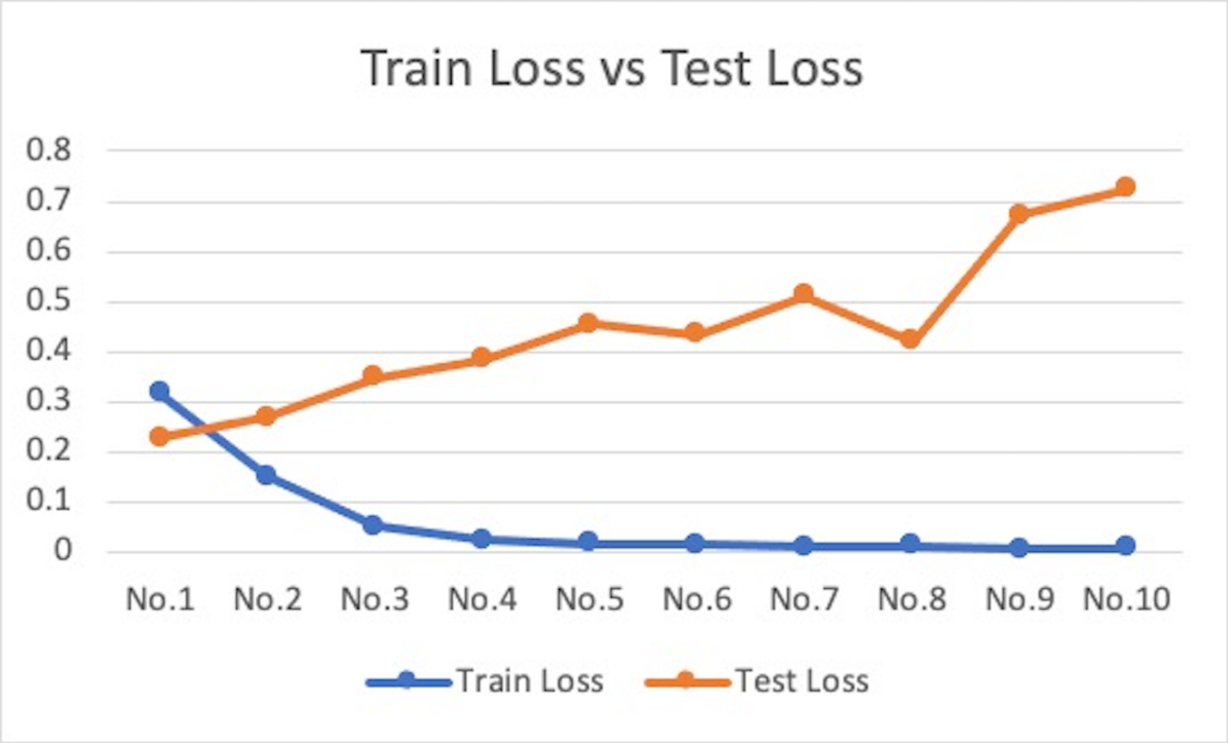
Homework 2

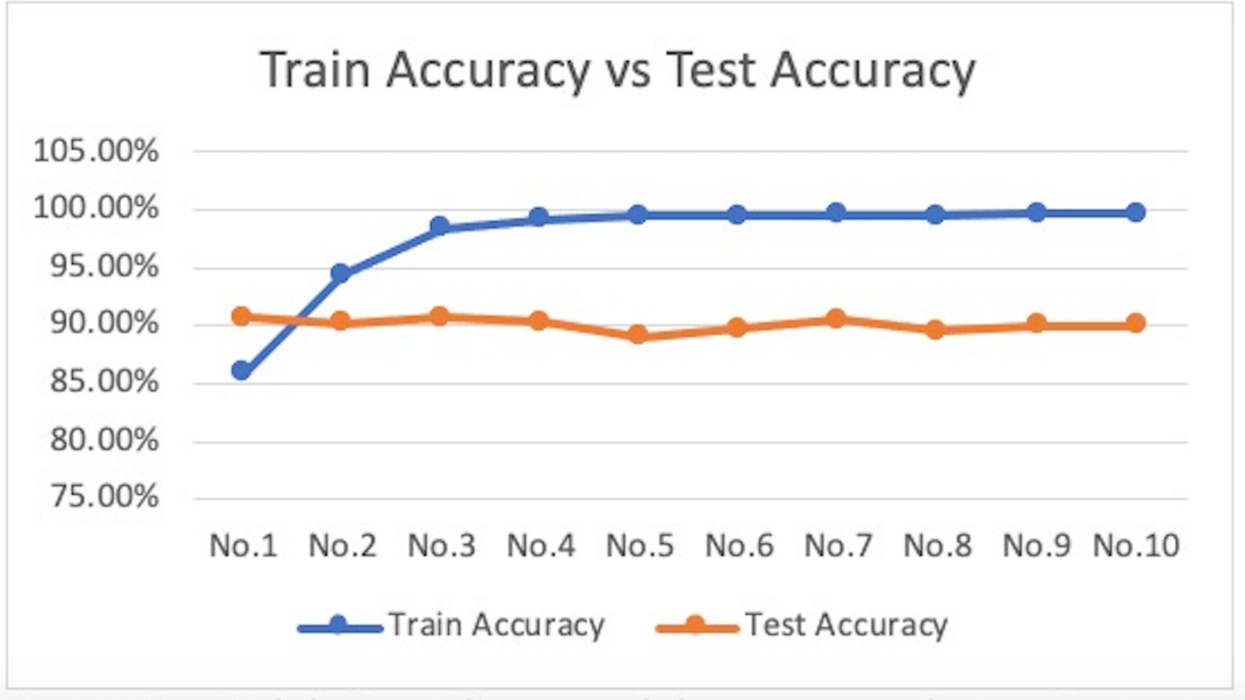
Problem 1:
Results:
Use finetuned-bert-model-12VA.pt

Table:

Epoch	Train Loss	Test Loss	Train Accuracy	Test Accuracy
1	0.316	0.229	85.96%	90.67%
2	0.153	0.270	94.44%	90.24%
3	0.052	0.349	98.37%	90.66%
4	0.026	0.384	99.21%	90.34%
5	0.018	0.456	99.46%	89.01%
6	0.015	0.434	99.48%	89.71%
7	0.012	0.512	99.61%	90.53%
8	0.013	0.421	99.49%	89.54%
9	0.008	0.673	99.72%	90.02%
10	0.010	0.725	99.66%	90.03%

Curves:





Challenges:

The first challenge I met is unable to upload whole file into google drive. For I download the data at my laptop, the webpage lost response every time. I solved this problem by using command in Colab to directly download data into google drive.

The second challenge is the program could not find models in saved model folder. Then, I follow the instruction of Readme.md file. After I upload the shared model, I finally make the program work in Colab.

Problem 2:

S1: Sales of the company to return to normalcy.

Grammars not in CNF:

PP	-->	IN DT NN

Grammars in CNF:

s	-->	NP	INF-VP	INF_VP	-->	TO	VP
NP	-->	NNS	PP	VP	-->	VB	PP
PP	-->	IN	NP	PP	-->	IN	NN
NP	-->	DT	NN				

S2: The new products and services contributed to increase revenue.

Grammars not in CNF:

NP	-->	DT JJ NNS CC. NNS

Grammars in CNF:

S	-->	NP	VP	NNS	-->	NNS	CCNNS
NP	-->	DT	AP	INF-VP	-->	TO	VP
AP	-->	JJ	NNS	VP	-->	VB	NN
CCNNS	-->	CC	NNS	VP	-->	VBD	INF-VP

S3: Dow falls as recession indicator flashed red and economical worries continue through the month.

Grammars not in CNF:

S	-->	S CC S

Grammars in CNF:

S	-->	AP	VP	NP	-->	NN	Nom
VP	-->	VP	PP	VP	-->	VBD	JJ
PP	-->	IN	S	VP	-->	VBP	PP
CCS	-->	CC	S	PP	-->	IN	NP
S	-->	S	CCS	NP	-->	DT	NN
AP	-->	JJ	NNS	NP	-->	VBZ	PP

S4: Figure skater lands historic quadruple jump in senior international competition at the 2019 World Figure Skating Championships on Day 3 but could only clinch a silver medal.

Grammars not in CNF:

S	-->	NP VP CC VP
VP	-->	VBZ NP PP PP PP
VP	-->	MD RB VP
NP	-->	JJ JJ NP
PP	-->	IN JJ JJ NN
PP	-->	IN DT CD NN NN NN NNS

PP	-->	IN NN LS

Grammars in CNF:

S	-->	NP	INF-VP	INF_VP	-->	TO	VP
NP	-->	NNS	PP	VP	-->	VB	PP
PP	-->	IN	NP	PP	-->	IN	NN
NP	-->	DT	NN				

Then I generate the following grammar for CNF:

S	-->	NP	INF-VP	NP	-->	NN	Nom
S	-->	NP	VP	NP	-->	NN	NN
S	-->	AP	VP	NP	-->	NP	PP
S	-->	S	CCS	NP	-->	JJ	AP
S	-->	NNP	VP	NP	-->	DT	CDNom
VP	-->	VB	PP	PP	-->	IN	NP
VP	-->	VB	NN	PP	-->	IN	NN
VP	-->	VBD	INF-VP	PP	-->	IN	S
VP	-->	VP	PP	PP	-->	IN	NNLS
VP	-->	VBD	JJ	AP	-->	JJ	NNS
VP	-->	VBP	PP	AP	-->	JJ	NN
VP	-->	VBZ	PP	Nom	-->	NN	NNS
VP	-->	VP	CCVP	Nom	-->	NN	Nom
VP	-->	VBZ	NP	NNS	-->	NNS	CCNNS
VP	-->	MDRB	VP	NNLS	-->	NN	LS
VP	-->	VP	NP	MDRB	-->	MD	RB
VP	-->	VB	NP	JJ	-->	JJ	JJ
INF-VP	-->	TO	VP	CDNom	-->	CD	Nom
NP	-->	NNS	PP	CCVP	-->	CC	VP
NP	-->	DT	NN	CCS	-->	CC	S
NP	-->	DT	AP	CCNNS	-->	CC	NNS
NNS	-->	sales|products|services | worries | championships		IN	-->	of|to|as|through |in|at|on	
JJ	-->	new|red|economical|historic |quadruple|senior|international|silver		NN	-->	company|normalcy|revenue|recession|indicator|month|figure|skater|jump|competition|world|skating|day|medal	
TO	-->	TO		DT	-->	the|a	
VB	-->	return|increase|clinch		CC	-->	and|but	
VBD	-->	contributed|flashed		NP	-->	Dow	
VBZ	-->	falls|lands		VBP	-->	continue	

LS	-->	3	CD	-->	2019
RB	-->	only	MD	-->	could

For program, suggested running code:

```
python3 hw2_CKYparser.py grammars.txt sentences.txt p1_output.txt
```

The output of programs is listed below: (also saved in p1_output.txt)

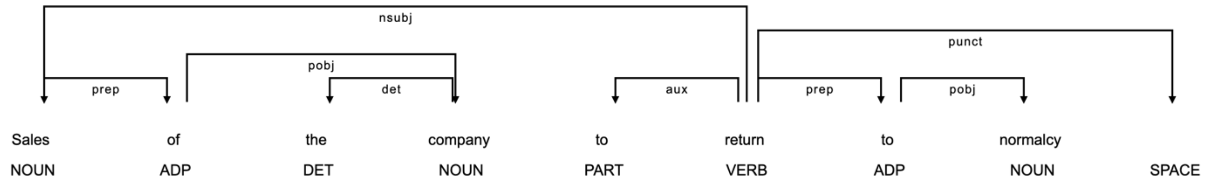
```
--S1--
# Sentence
Sales of the company to return to normalcy
# Bracketed structure parses
[NP Sales] [PP of] [NP the company] [INF-VP to] [VP return] [PP to normalcy]
# Num of parses
2

--S2--
# Sentence
The new products and services contributed to increase revenue
# Bracketed structure parses
[NP The] [AP new products] [CC and] [AP services] [VP contributed] [INF-VP to] [VP increase
revenue]
# Num of parses
1

--S3--
# Sentence
Dow falls as recession indicator flashed red and economical worries continue through the month
# Bracketed structure parses
[NP Dow] [VP falls] [PP as] [NP recession indicator] [VP flashed red] [CC and] [AP economical
worries] [VP continue] [PP through] [NP the month]
# Num of parses
2

--S4--
# Sentence
Figure skater lands historic quadruple jump in senior international competition at the 2019 World
Figure Skating Championships on Day 3 but could only clinch a silver medal
# Bracketed structure parses
[NP Figure skater] [VP lands] [NP historic] [AP quadruple jump] [PP in] [NP senior] [AP
international competition] [PP at] [NP the 2019 World Figure Skating Championships] [PP on
Day 3] [CC but] [VP could only clinch] [NP a] [AP silver medal]
# Num of parses
196
```

Problem 3:
I use Spacy to generate and visualize the dependency trees of four sentences in two kind of pipelines.
Code is provided by:
Problem3.py
I use PyCharm and execute it directly in IDE. The code could not automatically stop for generate a html. Manually terminate is required.

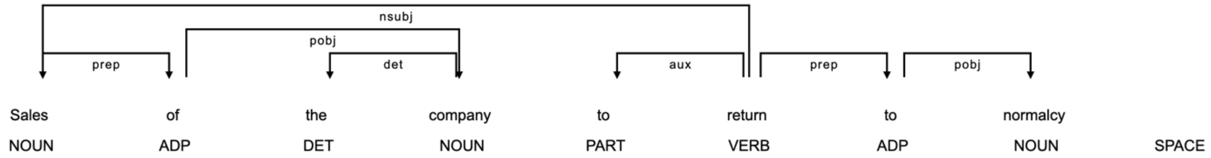
The images on webpage are long and narrow. I add screenshots below, but it is quite blur if the sentence is long. Hence, I still suggest you directly see here:
dependency trees.html

I manually add labels for each image in the above html file. The original version is:
dependency trees original.html

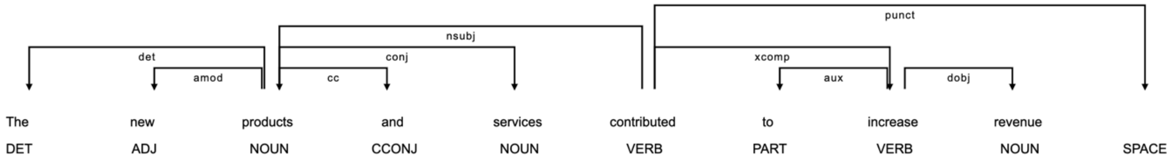
The sequence in webpage is just same as I am showing screenshots below:
S1 in en_core_web_sm



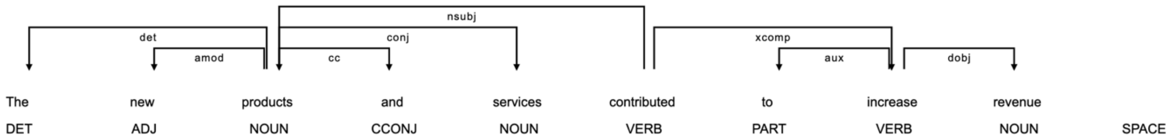
S1 in en_core_web_trf



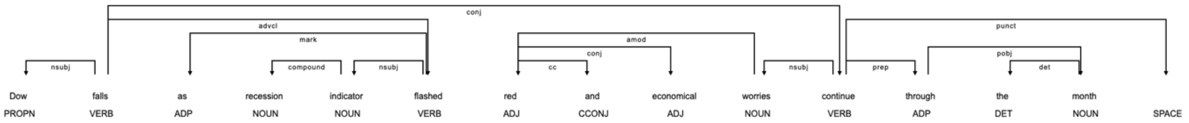
S2 in en_core_web_sm



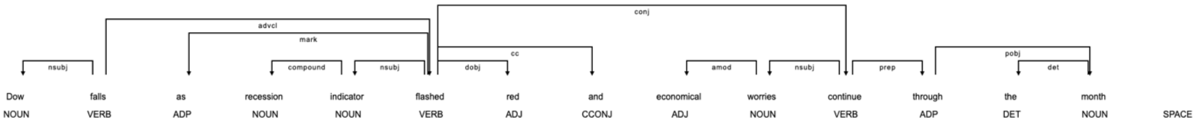
S2 in en_core_web_trf



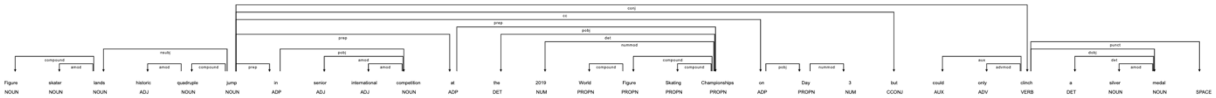
S3 in en_core_web_sm



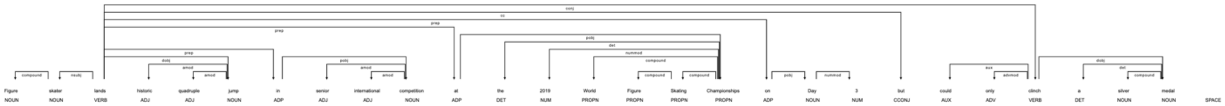
S3 in en_core_web_trf



S4 in en_core_web_sm

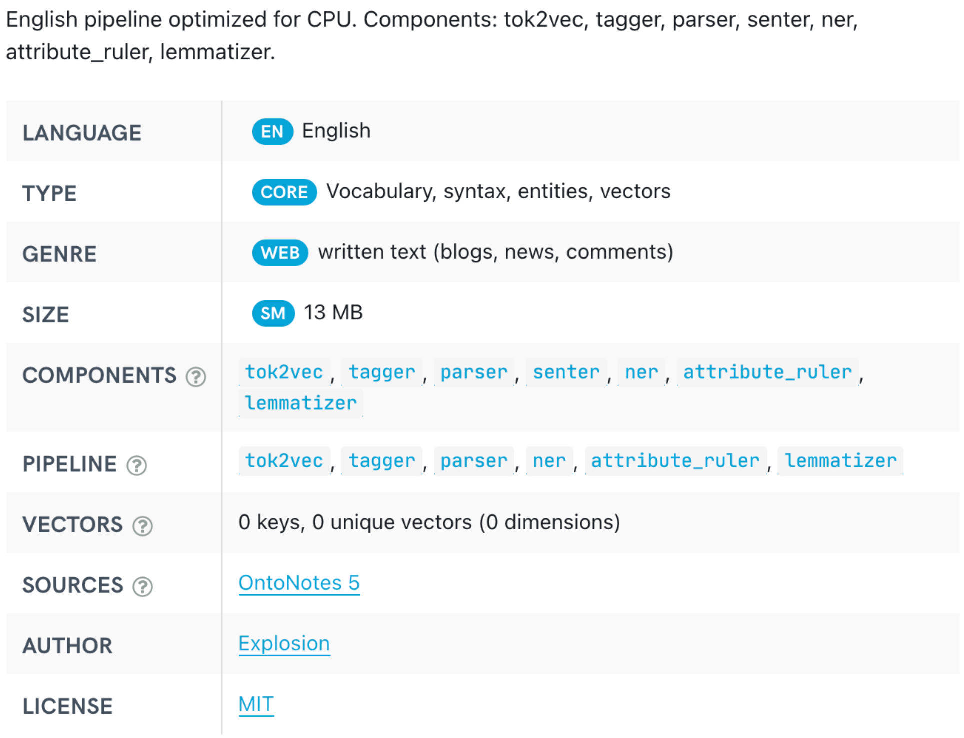


S4 in en_core_web_trf



From above images can we see, different pipelines will affect the results on the same sentence. For my observation, the dependency tree generate by en_core_web_sm is more complicated than by en_core_web_trf.

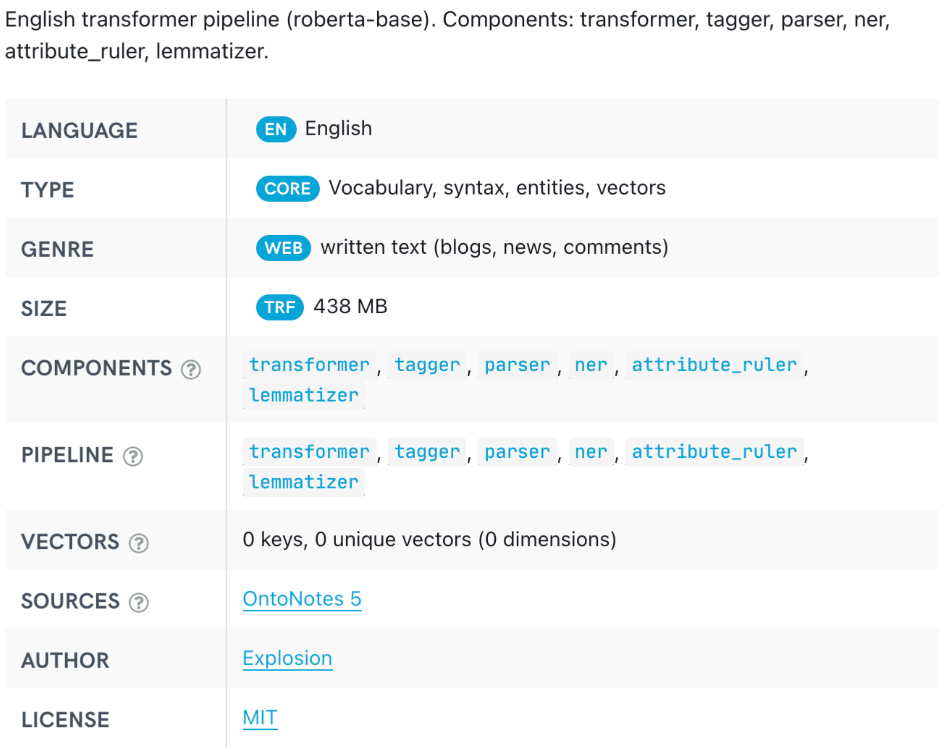
Below is some related material I found in spaCy's website.
en_core_web_sm:



English pipeline optimized for CPU. Components: tok2vec, tagger, parser, senter, ner, attribute_ruler, lemmatizer.

LANGUAGE	EN English
TYPE	CORE Vocabulary, syntax, entities, vectors
GENRE	WEB written text (blogs, news, comments)
SIZE	SM 13 MB
COMPONENTS	`tok2vec`, `tagger`, `parser`, `senter`, `ner`, `attribute_ruler`, `lemmatizer`
PIPELINE	`tok2vec`, `tagger`, `parser`, `ner`, `attribute_ruler`, `lemmatizer`
VECTORS	0 keys, 0 unique vectors (0 dimensions)
SOURCES	OntoNotes 5
AUTHOR	Explosion
LICENSE	MIT

en_core_web_trf:



English transformer pipeline (roberta-base). Components: transformer, tagger, parser, ner, attribute_ruler, lemmatizer.

LANGUAGE	EN English
TYPE	CORE Vocabulary, syntax, entities, vectors
GENRE	WEB written text (blogs, news, comments)
SIZE	TRF 438 MB
COMPONENTS	`transformer`, `tagger`, `parser`, `ner`, `attribute_ruler`, `lemmatizer`
PIPELINE	`transformer`, `tagger`, `parser`, `ner`, `attribute_ruler`, `lemmatizer`
VECTORS	0 keys, 0 unique vectors (0 dimensions)
SOURCES	OntoNotes 5
AUTHOR	Explosion
LICENSE	MIT