Requirements

In the project you will use a model to translate paragraphs from German to English.

For performance testing We will use the BLUE score(measure) which is a standard measure for measuring performance in translation tasks. The score will be calculated for each of the translated paragraphs, and the accuracy score will be the average BLEU scores received for translations of all paragraphs. The model must have an average BLEU score of at least 35% in file tagging val.labeled. When training on the train.labeled file only.

For the purposes of calculating the BLEU index, we will give you an example code named project_evaluate.py that we will use to calculate the performance of your models. Make sure that the code can run on the files you submit.

Compliance with the format: implementing a tagger, which receives a file in comp.unlabeld format, labels it and outputs a file named comp_id1_id2.labeld in the exact format.

You need to use transformers.

Data:

Explain of the attached files:

1. train.labeled: A file containing 10000 pairs of paragraphs in German and English. Each pair of sentences is separated by an empty line, and before each sentence there is a line containing the language in which it is written.

Example :

German:

Abdullah wollte das Treffen, weil er glaubt, dass das Weltgeschehen seit dem Jahre 2001 die Bruderschaft der Konservativen gespalten hat.

Bis dahin teilten er und Bush eine gemeinsame Weltsicht, die die Bedeutung der Religion, der traditionellen Familie (so, wie beider Länder sie auffassten), gesellschaftliche Disziplin und die Rolle des Staates als Unterstützer dieser Institutionen betonte.

English:

Abdullah sought the meeting because he believes that the world since 2001 has divided the fraternity of conservatives.

Until then, he and Bush shared a common worldview, emphasizing the importance of religion, the traditional family (as both countries understood it), social discipline, and the state's role in supporting these institutions.

German:

…

2. val.labeled: A file containing 1000 pairs of paragraphs in German and English, in the same format as train.labeled. You should use this file to evaluate the performance of your model.
3. comp.unlabled: A file containing 2000 paragraphs in German only. In addition, for each paragraph, the roots of each sentence in it appear, as well as two of the modifiers of the root, if any exist.

Example:
German:
Leider wurde in den zwei Jahren seit dem Zusammenbruch von Lehman kaum etwas getan, um dieses Risiko in Angriff zu nehmen.
Der US-Kongress ist dabei, einen Gesetzentwurf abzuschließen, der einem neu zu gründenden Rat für systemische Risiken die Befugnis zur Abwicklung großer US-Finanzinstitute einräumen soll.
Die Verfahren zur Auslösung dieser Intervention sind jedoch komplex, und die Finanzierung ist ausreichend vage geregelt, dass der Gesetzentwurf Kollateralschäden, die sich aus einem großen Bankenzusammenbruch ergeben, selbst für US-Institute nicht ausschließen wird – und für internationale Institute, deren Abwicklung die Koordinierung durch mehrere Staaten mit einem unterschiedlichen Grad an Solvenz erfordern würde, schon gar nicht.
Roots in English: done, is, are
Modifiers in English: (little, been), (about, Congress), (procedures, is)

German:
…

4. val.unlabeled: A file containing the sentences from the val file, in the same format as the comp file.

Train :
You can train any basic model (base size or large), do not use models trained for other tasks, especially not for translation tasks.
The competition file should not be used for any purpose at the training or validation stage except to run the model on it and save the results to the file.
You must train a model based on the tagged file train.labeled and save it to memory.

Inference :
You need to build a tagger that receives an untagged file and tags it using the model that will load from memory.
You must label the Val.unlabeled file based on a model that was trained on the train.labeled file only.
A Val_id1_id2.labeled file must be submitted in a format identical to the file Train.labeled.
The average BLEU scores received on the file Val_id1_id2.labeled should be reported in the report, compared to the file labeled Val.labeled.
Please note that accuracy should be reported compared to actual labeling.

Test :

You must label/tag comp.unlabeled. A comp_id1_id2.labeled file must be submitted in a format that is completely identical to the train.labeled file. Based on this tag, your competitive score will be determined.

Workspace:

The project must run on the machine given to you in the azureml_py38 environment without installing additional libraries. No additional libraries may be used without permission from the

course staff in the forum for the project. Do not use unprovisioned data files during course submissions.

Report :

Writing a detailed, concise, and to-the-point report (up to 3 pages) that will present the work. The report will include all the required sections and meet the submission conditions:

1. Description of experiments you performed during the work on the project.
2. Description of the algorithm you used to train the model and tag the Val file.
3. The percentage of accuracy (mean of the blue score like point out previously)obtained on the val file.
4. Description of experiments you performed to improve the results of the competition.
5. Description of the algorithm you used to train the competitive model and tag the comp file.
6. Expected accuracy of comp file.

More:

- The Files you tagged must be name Val_id1_id2.labeld and comp_id1_id2.labeld.
- The project code files must be documented and readable.
- Additionally, the code should be able to run on the virtual machine provided for the project. Please write simple running interfaces to train, test and generate the tagged competition files.
- An interface for tagging the competition files named generate_comp_tagged that receives a file in unlabeled format, labels it and outputs a file named comp_id1_id2.labeled in the same format as the train.labeled file format. The file should have a function that tags the val file and another that tags the comp file, and outputs each of them to a file with the appropriate name
- All the models that you create in order to resolve this task must be join with chosen name. (and brief explanation of each model in the report)
- Do not copy ready-made code snippets from the Internet, and in general do not rely on any other source of code other than your creation and the external packages specified in the relevant section.

FAQ

- We don't have the evaluate library installed on the azureml_py38_PT_and_TF environment even though it appears in the py file you gave us. Is it possible to install the library on the environment?

  it is possible

- We saw that there is the ROOT and the MODIFIER only in the UNLABELLED files, if we want to receive them as INPUT we should also find them in TRAIN but we don't

have them there. Do we also need to create models found in TRAIN to train our final model with them? Or do you not need to use them in TRAIN at all? We are not so sure that we understood how to use this information and it is not explained in the exercise.

If you want to use this information you need to create it. You can think of different ways to use it

- Is it allowed to use BertForMaskedLM and the built-in pad_sequences function of KERAS?

  You can use the different departments of huggingface, do not use keras.

- I think the work environment we got doesn't have torchtext, am I wrong?
  If not, can I install this library?

  You can install it.

- We chose to use the T5-base model that underwent pre-training only for the purpose of carrying out the project. From what we read about it, we saw that in order to perform fine tuning on it for the task of translation, we must add a prefix of the form: translate German to English
  Following the directive not to use models intended for translation, and even though the aforementioned model was not trained for translation, we wanted to make sure that it is indeed allowed to use this prefix.

  As we said in a previous comment, there is no problem using t5-base.
  Regarding the prefix - the use of text, usually at the beginning of the example, to direct the model to one task or another, is an idea called prompting and has become accepted in language processing in recent years. Note that you do not have to use this specific prompt to use t5 for translation. You can also think of different prompts that could help with the specific task more than the one offered in t5

- Is it possible to use pre trained embeddings of any kind?

  Yes
- In the submissions it is stated that "data files that were not provided during the submissions in the course must not be used". Is it possible to use the data of previous submissions if it seems appropriate?

  Yes

- Since we can use pre-trained models, I wanted to know if we can use this model https://huggingface.co/Helsinki-NLP/opus-mt-de-en for the tokenisation ?

  Do not use models trained on translation tasks. See response to using the pretrain model for further clarification.

- Can we consider a certain value as the maximum number of word in a sentence ? Or is our model have to be able to translate text of any possible length ?

  The competition files on which the code should run are given to you. You can think how to deal with the lengths.

- 1. Are we allowed to use any module of Hugging Face? in particular:

  A. In BLEU's prepared matrics from the Datasets library

  B. Use TRAINERS modules such as Seq2SeqTrainer.

  Are there certain modules that should not be used from Hugging Face?

  2. Are we allowed to use the Roots, Modifiers tagging in the VALIDATION data in order to optimize the performance of the model for the translation task?

  Yes to all

All the package that can be used to run in our servers :

Package	asn1crypto	backports.shutil-get-termi
Version	1.4.0	nal-size 1.0.0
---	astroid	backports.tempfile
---	2.5	1.0
alabaster	astropy	backports.weakref
0.7.12	4.2.1	1.0.post1
alembic	async-generator	beautifulsoup4
1.8.1	1.10	4.9.3
anaconda-client	atomicwrites	bitarray
1.7.2	1.4.0	2.1.0
anaconda-navigator	attrs	bkcharts
2.0.3	20.3.0	0.2
anaconda-project	autopage	black
0.9.1	0.5.1	19.10b0
anyio	autopep8	bleach
2.2.0	1.5.6	3.3.0
appdirs	Babel	bokeh
1.4.4	2.9.0	2.3.2
argh	backcall	boto
0.26.2	0.2.0	2.49.0
argon2-cffi	backports.functools-lru-ca	Bottleneck
20.1.0	che 1.6.4	1.3.2

brotlipy
0.7.0
certifi
2020.12.5
cffi
1.14.5
chardet
4.0.0
click
7.1.2
cliff
4.1.0
cloudpickle
1.6.0
clyent
1.2.2
cmaes
0.9.0
cmd2
2.4.2
colorama
0.4.4
colorlog
6.7.0
conda
4.10.1
conda-build
3.21.4
conda-content-trust
0+unknown
conda-package-handling
1.7.3
conda-repo-cli
1.0.4
conda-token
0.3.0
conda-verify
3.4.2
contextlib2
0.6.0.post1
cryptography
3.4.7
cycler
0.10.0
Cython
0.29.23
cytoolz
0.11.0
dask
2021.4.0
decorator
5.0.6
defusedxml
0.7.1
diff-match-patch
20200713
distributed
2021.4.1
docutils
0.17.1
entrypoints
0.3
et-xmlfile
1.0.1
fastcache
1.1.0
filelock
3.0.12
flake8
3.9.0
Flask
1.1.2
fsspec
0.9.0
future
0.18.2
gensim
4.2.0
gevent
21.1.2
glob2
0.7
gmpy2
2.0.8
greenlet
1.0.0
h5py
2.10.0
HeapDict
1.0.1
html5lib
1.1
idna
2.10
imageio
2.9.0
imagesize
1.2.0
importlib-metadata
4.13.0
importlib-resources
5.10.1
iniconfig
1.1.1
intervaltree
3.1.0
ipykernel
5.3.4
ipython
7.22.0
ipython-genutils
0.2.0
ipywidgets
7.6.3
isort
5.8.0
itsdangerous
1.1.0
jdcal
1.4.1
jedi
0.17.2
jeepney
0.6.0
Jinja2
2.11.3
joblib
1.0.1
json5
0.9.5
jsonschema
3.2.0
jupyter
1.0.0
jupyter-client
6.1.12
jupyter-console
6.4.0
jupyter-core
4.7.1
jupyter-packaging
0.7.12
jupyter-server
1.4.1

jupyterlab
3.0.14
jupyterlab-pygments
0.1.2
jupyterlab-server
2.4.0
jupyterlab-widgets
1.0.0
keyring
22.3.0
kiwisolver
1.3.1
lazy-object-proxy
1.6.0
libarchive-c
2.9
llvmlite
0.36.0
locket
0.2.1
lxml
4.6.3
Mako
1.2.4
MarkupSafe
1.1.1
matplotlib
3.3.4
mccabe
0.6.1
mistune
0.8.4
mkl-fft
1.3.0
mkl-random
1.2.1
mkl-service
2.3.0
mock
4.0.3
more-itertools
8.7.0
mpmath
1.2.1
msgpack
1.0.2
multipledispatch
0.6.0
mypy-extensions
0.4.3
navigator-updater
0.2.1
nbclassic
0.2.6
nbclient
0.5.3
nbconvert
6.0.7
nbformat
5.1.3
nest-asyncio
1.5.1
networkx
2.5
nltk
3.6.1
nose
1.3.7
notebook
6.3.0
numba
0.53.1
numexpr
2.7.3
numpy
1.20.1
numpydoc
1.1.0
olefile
0.46
openpyxl
3.0.7
optuna
3.0.4
packaging
20.9
pandas
1.2.4
pandocfilters
1.4.3
parso
0.7.0
partd
1.2.0
path
15.1.2
pathlib2
2.3.5
pathspec
0.7.0
patsy
0.5.1
pbr
5.11.0
pep8
1.7.1
pexpect
4.8.0
pickleshare
0.7.5
Pillow
8.2.0
pip
21.0.1
pkginfo
1.7.0
pluggy
0.13.1
ply
3.11
prettytable
3.5.0
prometheus-client
0.10.1
prompt-toolkit
3.0.17
psutil
5.8.0
ptyprocess
0.7.0
py
1.10.0
pycodestyle
2.6.0
pycosat
0.6.3
pycparser
2.20
pycurl
7.43.0.6
pydocstyle
6.0.0
pyerfa
1.7.3

pyflakes 2.2.0
Pygments 2.8.1
pylint 2.7.4
pyls-black 0.4.6
pyls-spyder 0.3.2
pyodbc 4.0.0-unsupported
pyOpenSSL 20.0.1
pyparsing 2.4.7
pyperclip 1.8.2
pyrsistent 0.17.3
PySocks 1.7.1
pytest 6.2.3
python-dateutil 2.8.1
python-jsonrpc-server 0.4.0
python-language-server 0.36.2
pytz 2021.1
PyWavelets 1.1.1
pyxdg 0.27
PyYAML 5.4.1
pyzmq 20.0.0
QDarkStyle 2.8.1
QtAwesome 1.0.2
qtconsole 5.0.3
QtPy 1.9.0
regex 2021.4.4
requests 2.25.1
rope 0.18.0
Rtree 0.9.7
ruamel-yaml-conda 0.15.100
scikit-image 0.18.1
scikit-learn 0.24.1
scipy 1.8.1
seaborn 0.11.1
SecretStorage 3.3.1
Send2Trash 1.5.0
setuptools 52.0.0.post20210125
simplegeneric 0.8.1
singledispatch 0.0.0
sip 4.19.13
six 1.15.0
smart-open 6.2.0
sniffio 1.2.0
snowballstemmer 2.1.0
sortedcollections 2.1.0
sortedcontainers 2.3.0
soupsieve 2.2.1
Sphinx 4.0.1
sphinxcontrib-applehelp 1.0.2
sphinxcontrib-devhelp 1.0.2
sphinxcontrib-htmlhelp 1.0.3
sphinxcontrib-jsmath 1.0.1
sphinxcontrib-qthelp 1.0.3
sphinxcontrib-serializinghtml 1.1.4
sphinxcontrib-websupport 1.2.4
spicy 0.16.0
spyder 4.2.5
spyder-kernels 1.10.2
SQLAlchemy 1.4.15
statsmodels 0.12.2
stevedore 4.1.1
sympy 1.8
tables 3.6.1
tblib 1.7.0
terminado 0.9.4
testpath 0.4.4
textdistance 4.2.1
threadpoolctl 2.1.0
three-merge 0.1.1
tifffile 2020.10.1
toml 0.10.2
toolz 0.11.1
torch 1.9.1+cpu

tornado
6.1
tqdm
4.59.0
traitlets
5.0.5
typed-ast
1.4.2
typing-extensions
3.7.4.3
ujson
4.0.2
unicodecsv
0.14.1
urllib3
1.26.4
watchdog
1.0.2
wcwidth
0.2.5
webencodings
0.5.1
Werkzeug
1.0.1
wheel
0.36.2
widgetsnbextension
3.5.1
wrapt
1.12.1
wurlitzer
2.1.0
xlrd
2.0.1
XlsxWriter
1.3.8
xlwt
1.3.0
xmltodict
0.12.0
yapf
0.31.0
zict
2.0.0
zipp
3.4.1
zope.event
4.5.0
zope.interface
5.3.0