

Appendix A

Extra Information

A.1 All models trained

A large number of models was trained as a result of this project. Most of the successful ones were uploaded to the [Huggingface Hub](#). This allows for easy access, backup, and allows other people to use them for their projects and research.

A complete list of all the models created and uploaded is below:

1. MikolajDeja/gsarti-opus-mt-tc-en-pl-kde4-finetune
2. MikolajDeja/gsarti-opus-mt-tc-en-pl-opus100-accelerate-finetune
3. MikolajDeja/gsarti-opus-mt-tc-en-pl-opus100-finetune
4. MikolajDeja/gsarti-opus-mt-tc-en-pl-opus100-finetune-100
5. MikolajDeja/gsarti-opus-mt-tc-en-pl-opus100-finetune-50
6. MikolajDeja/gsarti-opus-mt-tc-en-pl-3-para__crawl-finetune
7. MikolajDeja/gsarti-opus-mt-tc-en-pl-para__crawl-finetune
8. MikolajDeja/gsarti-opus-mt-tc-en-pl-yhavinga-ccmatrix-finetune
9. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-kde4-finetune
10. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-opus100-accelerate
11. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-opus100-finetune
12. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-opus100-finetune-100

13. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-opus100-finetune-50
14. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-3-para__crawl-finetune
15. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-para__crawl-finetune
16. MikolajDeja/Helsinki-NLP-opus-mt-pl-en-yhavinga-ccmatrix-finetune
17. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-kde4-finetune
18. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-opus100-accelerate
19. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-opus100-finetune
20. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-opus100-finetune-100
21. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-opus100-finetune-50
22. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-3-para__crawl-finetune
23. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-para__crawl-finetune
24. MikolajDeja/Helsinki-NLP-opus-mt-en-mul-yhavinga-ccmatrix-finetune
25. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-kde4-finetune
26. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-opus100-accelerate
27. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-opus100-finetune
28. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-opus100-finetune-100
29. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-opus100-finetune-50
30. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-3-para__crawl-finetune
31. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-para__crawl-finetune
32. MikolajDeja/Helsinki-NLP-opus-mt-mul-en-yhavinga-ccmatrix-finetune
33. MikolajDeja/alirezamsh-small100-en-pl-kde4-finetune
34. MikolajDeja/alirezamsh-small100-en-pl-opus100-accelerate-finetune
35. MikolajDeja/alirezamsh-small100-en-pl-opus100-finetune
36. MikolajDeja/alirezamsh-small100-en-pl-opus100-finetune-50

37. MikolajDeja/alirezamsh-small100-en-pl-3-para__crawl-finetune
38. MikolajDeja/alirezamsh-small100-en-pl-para__crawl-finetune
39. MikolajDeja/alirezamsh-small100-en-pl-yhavinga-ccmatrix-finetune
40. MikolajDeja/alirezamsh-small100-pl-en-kde4-finetune
41. MikolajDeja/alirezamsh-small100-pl-en-opus100-accelerate-finetune
42. MikolajDeja/alirezamsh-small100-pl-en-opus100-finetune
43. MikolajDeja/alirezamsh-small100-pl-en-opus100-finetune-50
44. MikolajDeja/alirezamsh-small100-pl-en-3-para__crawl-finetune
45. MikolajDeja/alirezamsh-small100-pl-en-para__crawl-finetune
46. MikolajDeja/alirezamsh-small100-pl-en-yhavinga-ccmatrix-finetune
47. MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-opus100-finetune
48. MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-3-para__crawl-finetune
49. MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-para__crawl
50. MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-yhavinga-ccmatrix-finetune
51. MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-opus100-finetune
52. MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-3-para__crawl-finetune
53. MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-para__crawl-finetune-copy
54. MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-yhavinga-ccmatrix-finetune

A.2 The design of the translator app

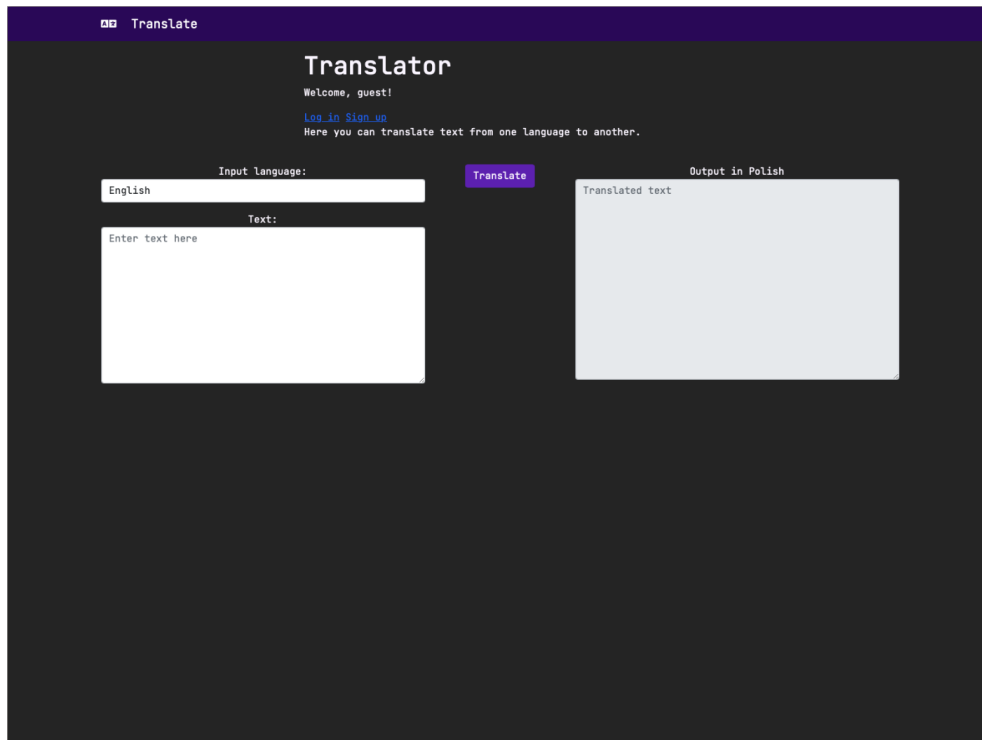


Figure A.1: The home page for a guest

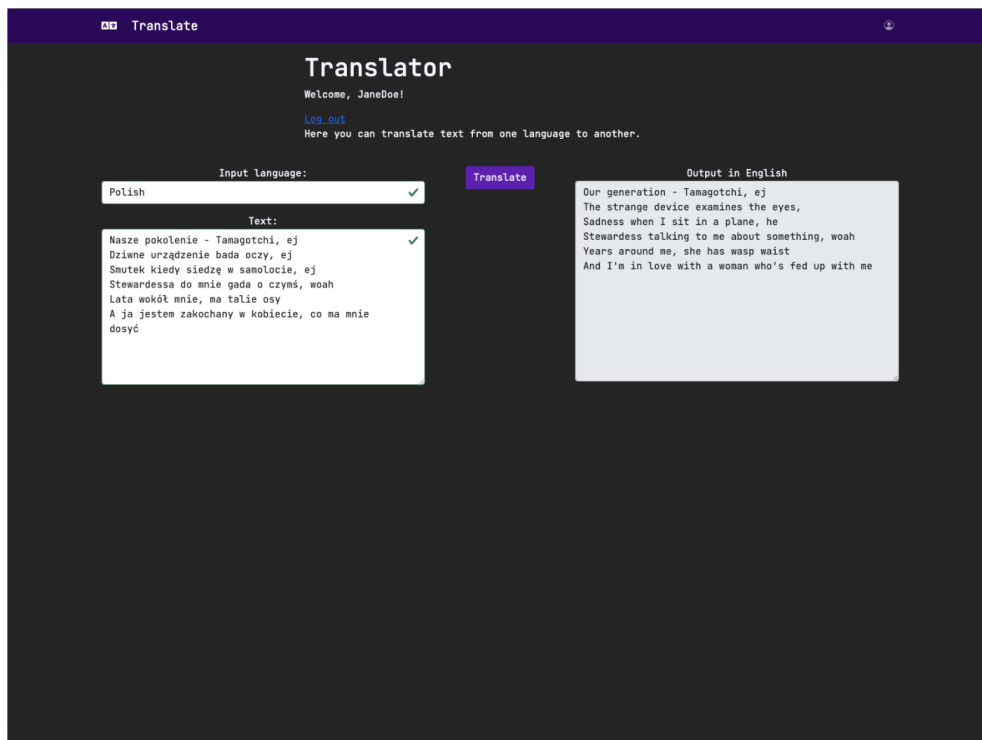


Figure A.2: The home page for a logged in user who translates from Polish to English

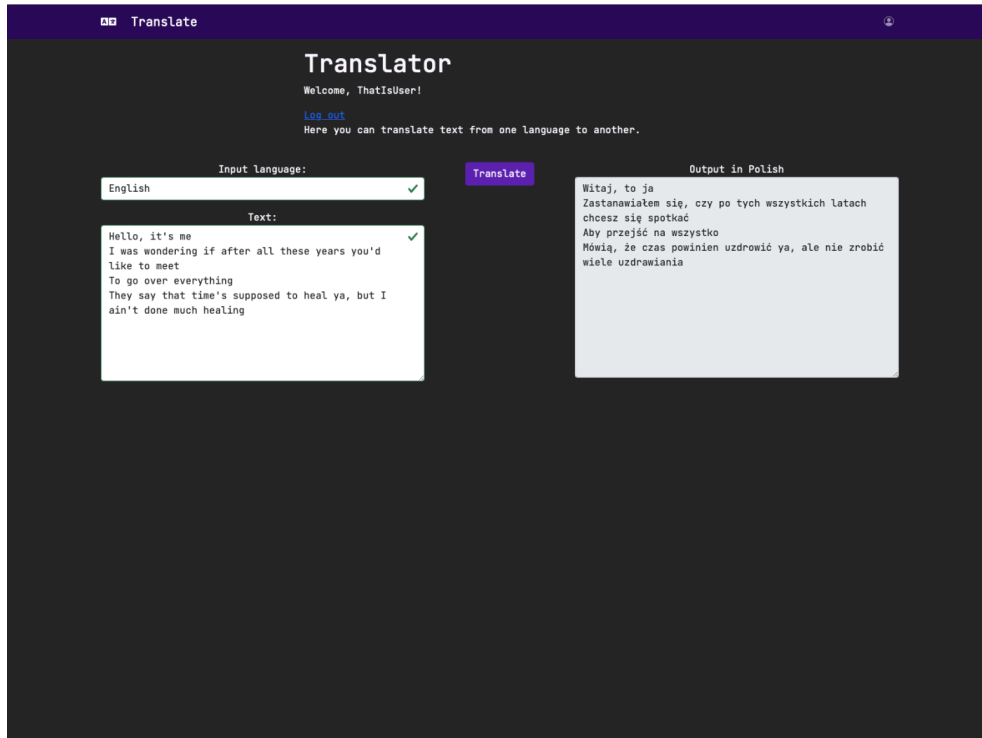


Figure A.3: The home page for a logged in user who translates from English to Polish

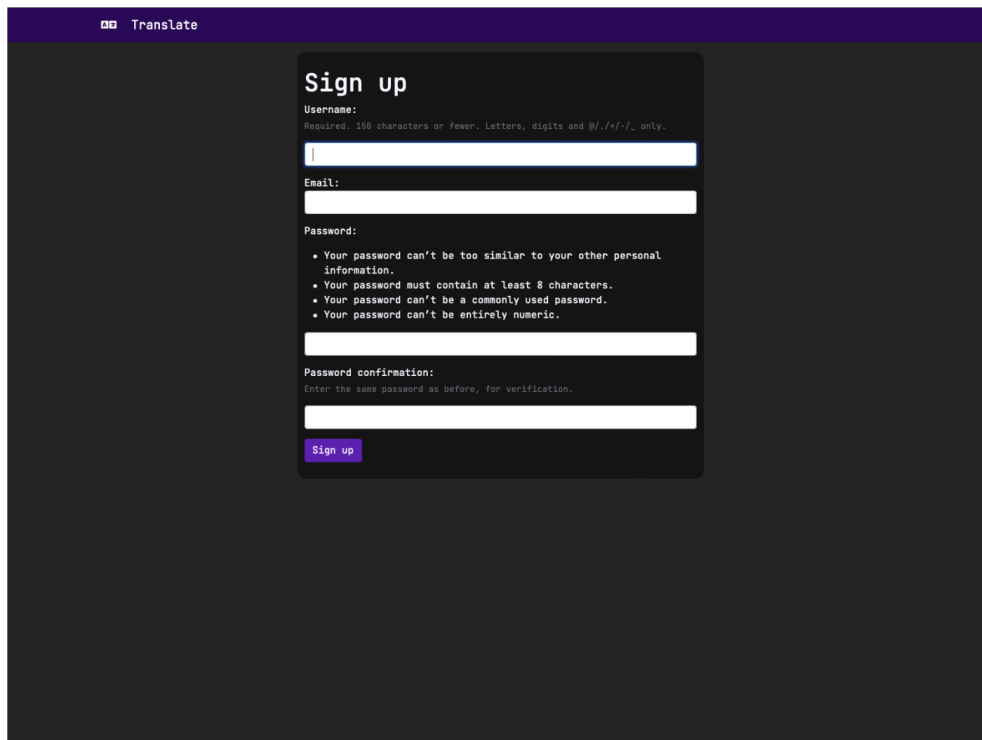


Figure A.4: The sign up page

Sign up

Username:
Required: 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Username

Email:
email@example.com

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation:
Enter the same password as before, for verification.

• This password is too common.

[Sign up](#)

Figure A.5: The sign up page with a password that is too common password

User profile

Username: JaneDoe
Email: janedoe@example.com

History of translations:

- Musze sorawić, żebyś zrozumiał → I must make you understand
- Chce ci tylko powiedzieć, I... → I just want to tell you how...
- You wouldn't get this from ... → Nie dostaniesz tego od innej...
- A full commitment's what I'... → Pełne zaangażowanie jest to...
- Znasz zasady i ja też (czy ja) → You know the rules and I to...
- We're no strangers to love → Nie jesteśmy obcymi dla miłości

Figure A.6: The profile page with the history of translations

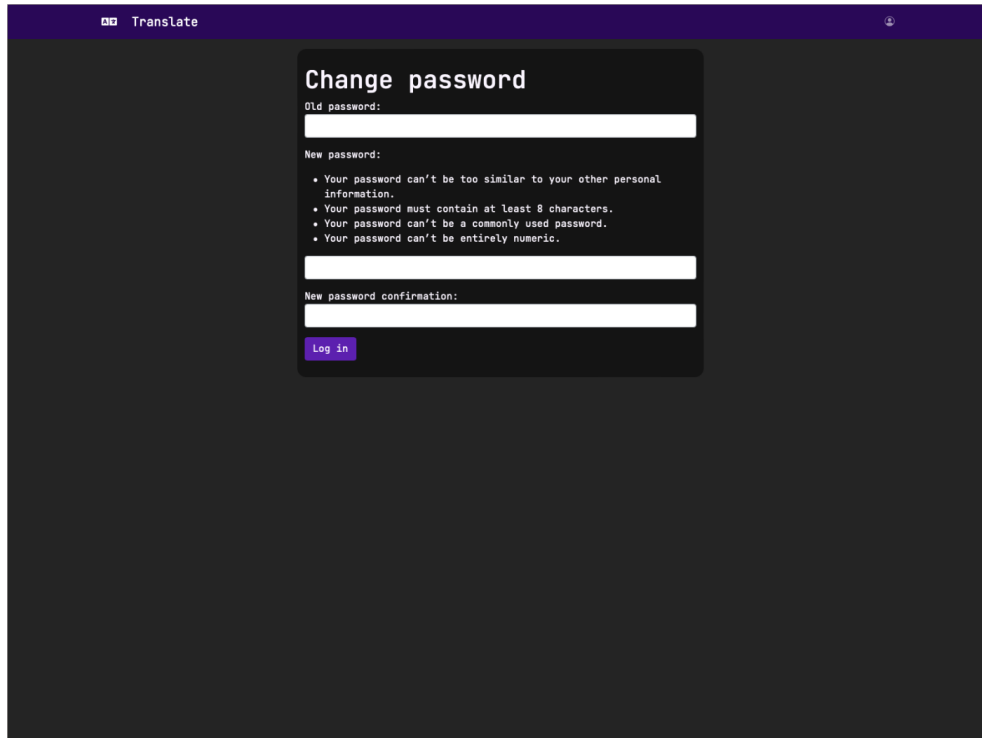


Figure A.7: The change password page

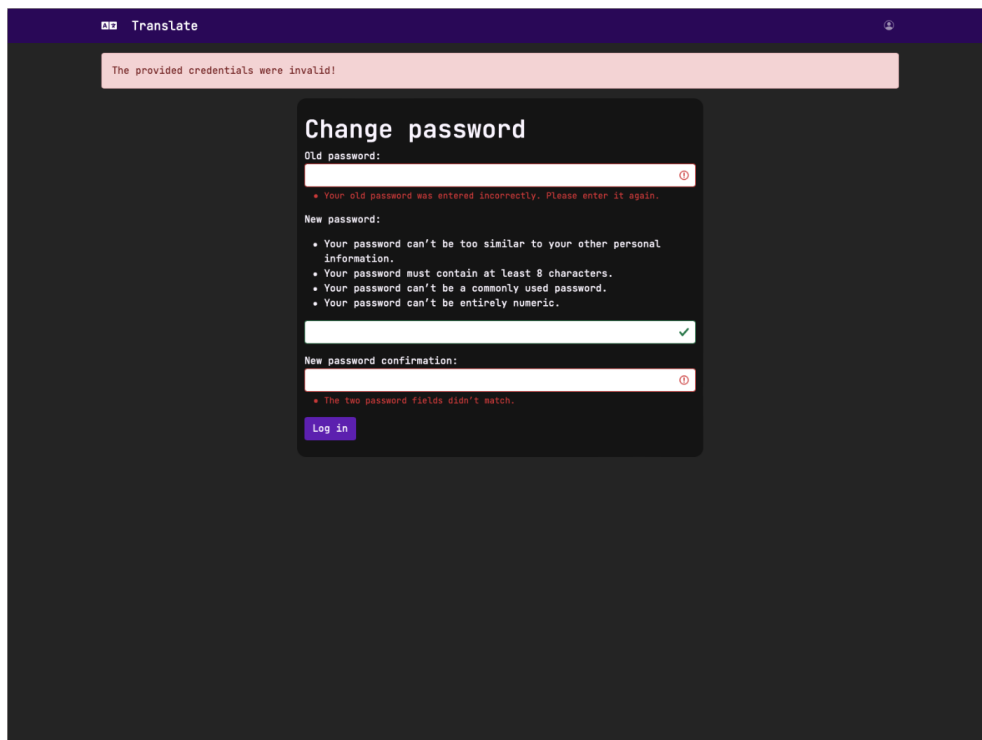


Figure A.8: The change password page with incorrect password confirmation

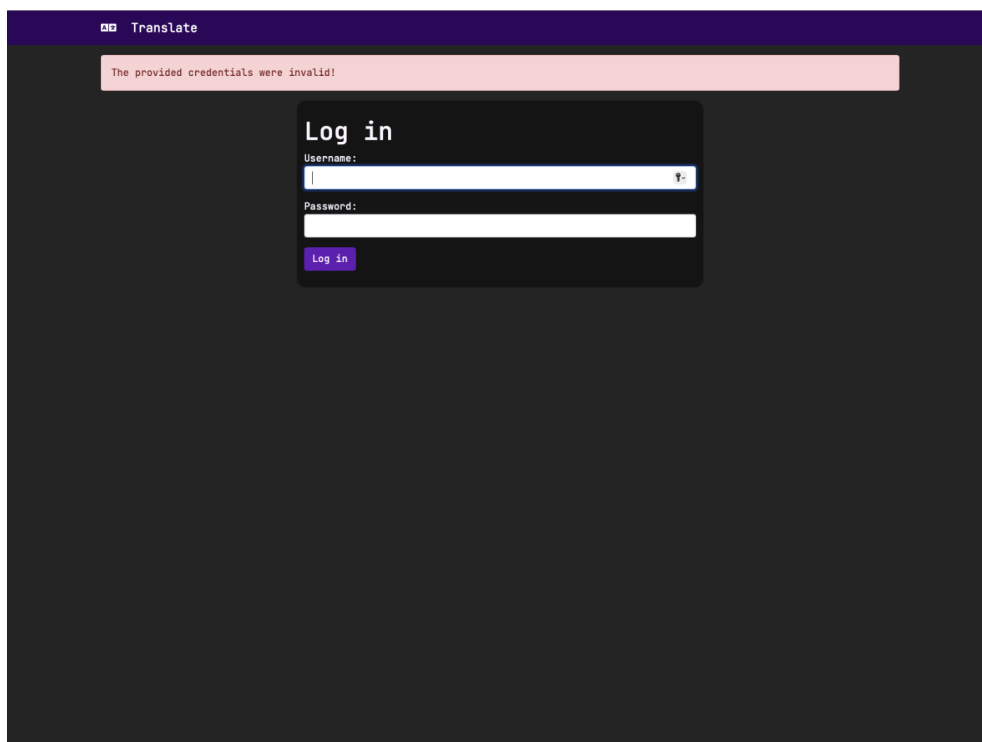


Figure A.9: Log in page with incorrect credentials

Appendix B

User Guide

B.1 Machine translation models

All of the models trained for this project (that were deemed valuable) were uploaded to the Huggingface Hub. A full list is available in Appendix A.1. All of them can easily be used with the Inference API.

```
pipeline_to_use = pipeline('translation', tokenizer='gsarti/opus-mt-tc-en-pl',
                             model='MikolajDeja/gsarti-opus-mt-tc-en-pl-3-para_crawl-finetune', src_lang='en',
                             )

pipeline_to_use = pipeline('translation', tokenizer='facebook/nllb-200-distilled-600M', model='MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-yhavinga-ccmatrix-finetune', src_lang='en',
                             tgt_lang='pl')
```

The pipelines for the M2M-100 based models require an additional argument of `tgt_lang`.

B.1.1 Training

Google Colaboratory

Should someone like to train the models in the same way that it was done for this project, a suitable environment must be prepared. Below, necessary packages are listed.

- `datasets`

- evaluate
- transformers[sentencepiece]
- sacrebleu

For uploading files to Huggingface Hub, `git-lfs` is required.

The above allows training on Google Colaboratory. In the Jupyter notebook, one can first add a code cell to install all the libraries and packages with `!apt install git-lfs` and `!pip install datasets evaluate transformers[sentencepiece] sacrebleu`. Then, the code in `training.py` should be copied to a code cell. After that, the chosen entry script can be copied to a code cell. This allows for a training to run.

HPC Create

For training on HPC Create, one must be more precise with the packages and modules loaded.

After literal weeks of fighting with it, I do not dare touch it and uninstall something needed.

The list of packages currently installed is:

<code>_libgcc_mutex</code>	0.1	main	
<code>_openmp_mutex</code>	5.1	1_gnu	
<code>absl-py</code>	1.4.0	pypi_0	pypi
<code>accelerate</code>	0.15.0	pypi_0	pypi
<code>aiohttp</code>	3.8.3	pypi_0	pypi
<code>aiosignal</code>	1.3.1	pypi_0	pypi
<code>alembic</code>	1.9.2	pypi_0	pypi
<code>anyio</code>	3.5.0	py38h06a4308_0	
<code>argon2-cffi</code>	21.3.0	pyhd3eb1b0_0	
<code>argon2-cffi-bindings</code>	21.2.0	py38h7f8727e_0	
<code>asttokens</code>	2.0.5	pyhd3eb1b0_0	
<code>async-timeout</code>	4.0.2	py38h06a4308_0	
<code>attrs</code>	22.1.0	py38h06a4308_0	
<code>babel</code>	2.11.0	py38h06a4308_0	
<code>backcall</code>	0.2.0	pyhd3eb1b0_0	
<code>beautifulsoup4</code>	4.11.1	py38h06a4308_0	
<code>blas</code>	1.0	mkl	
<code>bleach</code>	4.1.0	pyhd3eb1b0_0	
<code>bottleneck</code>	1.3.5	py38h7deecbd_0	
<code>brotlipy</code>	0.7.0	py38h27cfd23_1003	
<code>bzip2</code>	1.0.8	h7b6447c_0	
<code>c-ares</code>	1.18.1	h7f8727e_0	
<code>ca-certificates</code>	2023.01.10	h06a4308_0	
<code>cachetools</code>	5.3.0	pypi_0	pypi
<code>certifi</code>	2022.12.7	py38h06a4308_0	
<code>cffi</code>	1.15.1	py38h74dc2b5_0	
<code>charset-normalizer</code>	2.0.4	pyhd3eb1b0_0	
<code>click</code>	8.1.3	pypi_0	pypi
<code>cmes</code>	0.9.1	pypi_0	pypi
<code>colorama</code>	0.4.6	py38h06a4308_0	
<code>colorlog</code>	6.7.0	pypi_0	pypi
<code>comm</code>	0.1.2	py38h06a4308_0	
<code>cryptography</code>	38.0.4	py38h9ce1e76_0	
<code>cuda</code>	11.6.2	0	nvidia
<code>cuda-cccl</code>	11.6.55	hf6102b2_0	nvidia/label/cuda-11.6.2
<code>cuda-command-line-tools</code>	11.6.2	0	nvidia/label/cuda-11.6.2

cuda-compiler	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-cudart	11.6.55	he381448_0	nvidia/label/cuda-11.6.2
cuda-cudart-dev	11.6.55	h42ad0f4_0	nvidia/label/cuda-11.6.2
cuda-cuobjdump	11.6.124	h2eeebcb_0	nvidia/label/cuda-11.6.2
cuda-cupti	11.6.124	h86345e5_0	nvidia/label/cuda-11.6.2
cuda-cuxxfilt	11.6.124	hecbf4f6_0	nvidia/label/cuda-11.6.2
cuda-driver-dev	11.6.55	0	nvidia/label/cuda-11.6.2
cuda-gdb	11.6.124	hcdc6958_0	nvidia/label/cuda-11.6.2
cuda-libraries	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-libraries-dev	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-memcheck	11.6.124	ha4ac6c0_0	nvidia/label/cuda-11.6.2
cuda-nsight	11.6.124	0	nvidia/label/cuda-11.6.2
cuda-nsight-compute	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-nvcc	11.6.124	hbba6d2d_0	nvidia/label/cuda-11.6.2
cuda-nvdisasm	11.6.124	h75ac146_0	nvidia/label/cuda-11.6.2
cuda-nvml-dev	11.6.55	haa9ef22_0	nvidia/label/cuda-11.6.2
cuda-nvprof	11.6.124	h7c7a4e2_0	nvidia/label/cuda-11.6.2
cuda-nvprune	11.6.124	he22ec0a_0	nvidia/label/cuda-11.6.2
cuda-nvrtc	11.6.124	h020bade_0	nvidia/label/cuda-11.6.2
cuda-nvrtc-dev	11.6.124	h249d397_0	nvidia/label/cuda-11.6.2
cuda-nvtx	11.6.124	h0630a44_0	nvidia/label/cuda-11.6.2
cuda-nvvp	11.6.124	h38ac01c_0	nvidia/label/cuda-11.6.2
cuda-runtime	11.6.2	0	nvidia
cuda-samples	11.6.101	h8efea70_0	nvidia/label/cuda-11.6.2
cuda-sanitizer-api	11.6.124	h3d04c53_0	nvidia/label/cuda-11.6.2
cuda-toolkit	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-tools	11.6.2	0	nvidia/label/cuda-11.6.2
cuda-visual-tools	11.6.2	0	nvidia/label/cuda-11.6.2
datasets	2.10.1	pypi_0	pypi
debugpy	1.5.1	py38h295c915_0	
decorator	5.1.1	pyhd3eb1b0_0	
defusedxml	0.7.1	pyhd3eb1b0_0	
deprecated	1.2.13	py38h06a4308_0	
dill	0.3.6	pypi_0	pypi
distlib	0.3.6	pypi_0	pypi
entrypoints	0.4	py38h06a4308_0	
evaluate	0.4.0	pypi_0	pypi
executing	0.8.3	pyhd3eb1b0_0	
ffmpeg	4.2.2	h20bf706_0	
filelock	3.9.0	py38h06a4308_0	
flit-core	3.6.0	pyhd3eb1b0_0	
freetype	2.12.1	h4a9f257_0	
frozenlist	1.3.3	pypi_0	pypi
fsspec	2022.11.0	pypi_0	pypi
gds-tools	1.2.1.4	0	nvidia/label/cuda-11.6.2
giflib	5.2.1	h5eee18b_1	
gmp	6.2.1	h295c915_3	
gnutls	3.6.15	he1e5248_0	
google-auth	2.16.0	pypi_0	pypi
google-auth-oauthlib	0.4.6	pypi_0	pypi
greenlet	2.0.2	pypi_0	pypi
grpcio	1.51.1	pypi_0	pypi
huggingface-hub	0.11.1	pypi_0	pypi
icu	58.2	he6710b0_3	
idna	3.4	py38h06a4308_0	
importlib-metadata	4.11.3	py38h06a4308_0	
importlib-resources	5.2.0	pyhd3eb1b0_1	
intel-openmp	2021.4.0	h06a4308_3561	
ipykernel	6.19.2	py38hb070fc8_0	
ipython	8.8.0	py38h06a4308_0	
ipython_genutils	0.2.0	pyhd3eb1b0_1	
ipywidgets	8.0.4	pypi_0	pypi
jedi	0.18.1	py38h06a4308_1	
jinja2	3.1.2	py38h06a4308_0	
jpeg	9e	h7f8727e_0	
json5	0.9.6	pyhd3eb1b0_0	

jsonschema	4.16.0	py38h06a4308_0	
jupyter_client	7.4.8	py38h06a4308_0	
jupyter_core	5.1.1	py38h06a4308_0	
jupyter_server	1.23.4	py38h06a4308_0	
jupyterlab	3.5.2	pyhd8ed1ab_0	conda-forge
jupyterlab-widgets	3.0.5	pypi_0	pypi
jupyterlab_pygments	0.1.2	py_0	
jupyterlab_server	2.16.5	py38h06a4308_0	
jupyterlab_widgets	1.0.0	pyhd3eb1b0_1	
lame	3.100	h7b6447c_0	
lcms2	2.12	h3be6417_0	
ld_impl_linux-64	2.38	h1181459_1	
lerc	3.0	h295c915_0	
libcublas	11.9.2.110	h5e84587_0	nvidia/label/cuda-11.6.2
libcublas-dev	11.9.2.110	h5c901ab_0	nvidia/label/cuda-11.6.2
libcufft	10.7.2.124	h4fbf590_0	nvidia/label/cuda-11.6.2
libcufft-dev	10.7.2.124	h98a8f43_0	nvidia/label/cuda-11.6.2
libcufile	1.2.1.4	0	nvidia/label/cuda-11.6.2
libcufile-dev	1.2.1.4	0	nvidia/label/cuda-11.6.2
libcurand	10.2.9.124	h37c27f7_0	nvidia/label/cuda-11.6.2
libcurand-dev	10.2.9.124	h54cfa4b_0	nvidia/label/cuda-11.6.2
libcusolver	11.3.4.124	h33c3c4e_0	nvidia/label/cuda-11.6.2
libcusolver-dev	11.3.4.124	h203c794_0	nvidia/label/cuda-11.6.2
libcuspars	11.7.2.124	h7538f96_0	nvidia/label/cuda-11.6.2
libcuspars-dev	11.7.2.124	hbb9722_0	nvidia/label/cuda-11.6.2
libdeflate	1.8	h7f8727e_5	
libffi	3.3	he6710b0_2	
libgcc-ng	11.2.0	h1234567_1	
libgomp	11.2.0	h1234567_1	
libidn2	2.3.2	h7f8727e_0	
libnpp	11.6.3.124	hd2722f0_0	nvidia/label/cuda-11.6.2
libnpp-dev	11.6.3.124	h3c42840_0	nvidia/label/cuda-11.6.2
libnvjpeg	11.6.2.124	hd473ad6_0	nvidia/label/cuda-11.6.2
libnvjpeg-dev	11.6.2.124	hb5906b9_0	nvidia/label/cuda-11.6.2
libopus	1.3.1	h7b6447c_0	
libpng	1.6.37	hbc83047_0	
libprotobuf	3.20.3	he621ea3_0	
libsodium	1.0.18	h7b6447c_0	
libstdcxx-ng	11.2.0	h1234567_1	
libtasn1	4.16.0	h27cfd23_0	
libtiff	4.5.0	h6a678d5_1	
libunistring	0.9.10	h27cfd23_0	
libvpx	1.7.0	h439df22_0	
libwebp	1.2.4	h11a3e52_0	
libwebp-base	1.2.4	h5eee18b_0	
libxml2	2.9.14	h74e7548_0	
libxslt	1.1.35	h4e12654_0	
lxml	4.9.2	pypi_0	pypi
lz4-c	1.9.4	h6a678d5_0	
mako	1.2.4	pypi_0	pypi
markdown	3.4.1	pypi_0	pypi
markupsafe	2.1.1	py38h7f8727e_0	
matplotlib-inline	0.1.6	py38h06a4308_0	
mistune	0.8.4	py38h7b6447c_1000	
mkl	2021.4.0	h06a4308_640	
mkl-service	2.4.0	py38h7f8727e_0	
mkl_fft	1.3.1	py38hd3c417e_0	
mkl_random	1.2.2	py38h51133e4_0	
msgpack	1.0.4	pypi_0	pypi
msgpack-python	1.0.3	py38hd09550d_0	
multidict	6.0.4	pypi_0	pypi
multiprocess	0.70.14	pypi_0	pypi
nbclassic	0.4.8	py38h06a4308_0	
nbclient	0.5.13	py38h06a4308_0	
nbconvert	6.5.4	py38h06a4308_0	
nbformat	5.7.0	py38h06a4308_0	

ncurses	6.4	h6a678d5_0	
nest-asyncio	1.5.6	py38h06a4308_0	
nettle	3.7.3	hbbd107a_1	
notebook	6.5.2	py38h06a4308_0	
notebook-shim	0.2.2	py38h06a4308_0	
nsight-compute	2022.1.1.2	0	nvidia/label/cuda-11.6.2
numexpr	2.8.4	py38he184ba9_0	
numpy	1.23.5	py38h14f4228_0	
numpy-base	1.23.5	py38h31eccc5_0	
oauthlib	3.2.2	pypi_0	pypi
openh264	2.1.1	h4ff587b_0	
openssl	1.1.1s	h7f8727e_0	
optuna	3.1.0	pypi_0	pypi
packaging	22.0	py38h06a4308_0	
pandas	1.5.2	pypi_0	pypi
pandocfilters	1.5.0	pyhd3eb1b0_0	
parso	0.8.3	pyhd3eb1b0_0	
pexpect	4.8.0	pyhd3eb1b0_3	
pickleshare	0.7.5	pyhd3eb1b0_1003	
pillow	9.3.0	py38h6a678d5_2	
pip	22.3.1	py38h06a4308_0	
pkgutil-resolve-name	1.3.10	py38h06a4308_0	
platformdirs	2.5.2	py38h06a4308_0	
portalocker	2.6.0	pypi_0	pypi
prometheus_client	0.14.1	py38h06a4308_0	
prompt-toolkit	3.0.36	py38h06a4308_0	
protobuf	3.20.1	pypi_0	pypi
psutil	5.9.4	pypi_0	pypi
ptyprocess	0.7.0	pyhd3eb1b0_2	
pure_eval	0.2.2	pyhd3eb1b0_0	
pyarrow	10.0.1	pypi_0	pypi
pyasn1	0.4.8	pypi_0	pypi
pyasn1-modules	0.2.8	pypi_0	pypi
pycparser	2.21	pyhd3eb1b0_0	
pygments	2.11.2	pyhd3eb1b0_0	
pyopenssl	22.0.0	pyhd3eb1b0_0	
pyrsistent	0.18.0	py38heec7806_0	
pysocks	1.7.1	py38h06a4308_0	
python	3.8.12	h12debd9_0	
python-dateutil	2.8.2	pyhd3eb1b0_0	
python-fastjsonschema	2.16.2	py38h06a4308_0	
pytorch	1.13.1	py3.8_cuda11.6_cudnn8.3.2_0	pytorch
pytorch-cuda	11.6	h867d48c_0	pytorch
pytorch-mutex	1.0	cuda	pytorch
pytz	2022.7	py38h06a4308_0	
pyyaml	6.0	py38h5eee18b_1	
pyzmq	23.2.0	py38h6a678d5_0	
ray	2.2.0	pypi_0	pypi
ray-core	1.6.0	py38h295c915_0	
ray-tune	1.6.0	py38h06a4308_0	
readline	8.2	h5eee18b_0	
redis-py	4.3.4	py38h06a4308_0	
regex	2022.10.31	pypi_0	pypi
requests	2.28.1	py38h06a4308_0	
requests-oauthlib	1.3.1	pypi_0	pypi
responses	0.18.0	pypi_0	pypi
rsa	4.9	pypi_0	pypi
sacrebleu	2.3.1	pypi_0	pypi
scipy	1.10.0	pypi_0	pypi
send2trash	1.8.0	pyhd3eb1b0_1	
sentencepiece	0.1.97	pypi_0	pypi
setproctitle	1.1.10	py38h7b6447c_1001	
setuptools	65.6.3	py38h06a4308_0	
six	1.16.0	pyhd3eb1b0_1	
sniffio	1.2.0	py38h06a4308_1	
soupsieve	2.3.2.post1	py38h06a4308_0	

sqlalchemy	2.0.1	pypi_0	pypi
sqlite	3.40.1	h5082296_0	
stack_data	0.2.0	pyhd3eb1b0_0	
tabulate	0.9.0	pypi_0	pypi
tensorboard	2.11.2	pypi_0	pypi
tensorboard-data-server	0.6.1	pypi_0	pypi
tensorboard-plugin-wit	1.8.1	pypi_0	pypi
tensorboardx	2.5.1	pypi_0	pypi
terminado	0.17.1	py38h06a4308_0	
tinycss2	1.2.1	py38h06a4308_0	
tk	8.6.12	h1ccaba5_0	
tokenizers	0.13.2	pypi_0	pypi
tomli	2.0.1	py38h06a4308_0	
torchaudio	0.13.1	py38_cu116	pytorch
torchvision	0.14.1	py38_cu116	pytorch
tornado	6.2	py38h5eee18b_0	
tqdm	4.64.1	pypi_0	pypi
traitlets	5.7.1	py38h06a4308_0	
transformers	4.25.1	pypi_0	pypi
typing-extensions	4.4.0	py38h06a4308_0	
typing_extensions	4.4.0	py38h06a4308_0	
urllib3	1.26.14	py38h06a4308_0	
virtualenv	20.17.1	pypi_0	pypi
wcwidth	0.2.5	pyhd3eb1b0_0	
webencodings	0.5.1	py38_1	
websocket-client	0.58.0	py38h06a4308_4	
werkzeug	2.2.2	pypi_0	pypi
wheel	0.37.1	pyhd3eb1b0_0	
widetsnbextension	4.0.5	pypi_0	pypi
wrapt	1.14.1	py38h5eee18b_0	
x264	1!157.20191217	h7b6447c_0	
xxhash	3.2.0	pypi_0	pypi
xz	5.2.10	h5eee18b_1	
yaml	0.2.5	h7b6447c_0	
yaml	1.8.2	pypi_0	pypi
zeromq	4.3.4	h2531618_0	
zipp	3.11.0	py38h06a4308_0	
zlib	1.2.13	h5eee18b_0	
zstd	1.5.2	ha4553b6_0	

Admittedly, **Jupyter** was also installed, since at the beginning it was used for training. Also, packages for hyperparameter search backend are installed and they are not necessary for training. The important pieces are the **Huggingface** packages, **torch**, and the **cuda** packages for running on the Nvidia GPUs and all of their dependencies.

To run the training scripts on HPC Create, the helper bash scripts have to be used to submit jobs with **sbatch**. To run one particular training, the **train.sh** script should be edited and all the other entry level scripts should be commented out. Only one line with **python scripts/file.py** should not be commented out.

B.1.2 Hyperparameter search

The setup for hyperparameter search is very similar to that of training, the only difference is that the hyperparameter search backend, **optuna** must be installed.

B.1.3 Testing

For testing, fewer packages are needed.

- `transformers`
- `datasets`
- `evaluate`

The scripts can be run in an analogous way to the training scripts. Install the packages, copy the contents of `testing.py` and the entry level script to Google Colaboratory to run it there. Use the `test.sh` to run on HPC Create.

B.2 Practical application

Running the practical application locally is the same as running any Django application.

The following script will install the necessary packages, create database migrations and run the server locally. For this project, Python 3 is required (3.9 recommended).

```
virtualenv venv --python=3.9
source venv/bin/activate
pip3 install -r requirements.txt
python3 manage.py migrate
python3 manage.py runserver
```

After having ran this set of commands, the user can navigate to *localhost:8000* to access the application.

To run the tests, use

```
python manage.py test
```

To then see the test coverage, run

```
coverage run manage.py test
coverage report
```

Appendix C

Source Code

1. Machine learning

- (a) Training - the training scripts
- (b) Hyper - the scripts used for hyperparameter search
- (c) Testing - the testing scripts
- (d) HelperScripts - the scripts made for running the other scripts on HPC Create

2. Translator - the code for the practical application

I verify that I am the sole author of the programs contained in this folder, except where explicitly stated to the contrary.

Mikolaj Deja,

April 6, 2023

43 files

Training/TrainAlirezaEnPlCC.py
Training/TrainAlirezaEnPlOpus.py
Training/TrainAlirezaEnPlOpusAcc.py
Training/TrainAlirezaEnPlPara.py
Training/TrainAlirezaEnPlPara3.py
Training/TrainAlirezaEnPlkde4.py
Training/TrainAlirezaPlEnCC.py
Training/TrainAlirezaPlEnOpus.py
Training/TrainAlirezaPlEnOpusAcc.py
Training/TrainAlirezaPlEnPara.py
Training/TrainAlirezaPlEnPara3.py
Training/TrainAlirezaPlEnkde4.py
Training/TrainEnMulCC.py
Training/TrainEnMulOpus.py
Training/TrainEnMulOpusAcc.py
Training/TrainEnMulPara.py
Training/TrainEnMulPara3.py
Training/TrainEnMulKde4.py
Training/TrainEnPlCC.py
Training/TrainEnPlOpus.py
Training/TrainEnPlOpusAcc.py
Training/TrainEnPlPara.py
Training/TrainEnPlPara3.py
Training/TrainEnPlkde4.py
Training/TrainMulEnCC.py
Training/TrainMulEnOpus.py
Training/TrainMulEnOpusAcc.py
Training/TrainMulEnPara.py
Training/TrainMulEnPara3.py
Training/TrainMulEnkde4.py
Training/TrainNLLBEnPlCC.py
Training/TrainNLLBEnPlOpus.py
Training/TrainNLLBEnPlPara3.py
Training/TrainNLLBPlEnCC.py
Training/TrainNLLBPlEnOpus.py
Training/TrainNLLBPlEnPara3.py
Training/TrainPlEnCC.py
Training/TrainPlEnOpus.py
Training/TrainPlEnOpusAcc.py
Training/TrainPlEnPara.py
Training/TrainPlEnPara3.py
Training/TrainPlEnkde4.py
Training/training.py

Training/TrainAlirezaEnPlCC.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'yhavinga/ccmatrix'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
```

```

11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17 tgt_lang='pl')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['en'] for ex in examples['translation']]
23     targets = [ex['pl'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainAlirezaEnPlOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='pl')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,

```

```
30 | EPOCH_NUM, tokenize_help, max_length)
```

Training/TrainAlirezaEnPlOpusAcc.py

```
1 | from datasets import load_dataset
2 | from transformers import AutoTokenizer
3 |
4 | from training import train_accelerate
5 |
6 | DATASET = 'opus100'
7 | EPOCH_NUM = 50
8 | max_length = 128
9 | MODEL_NAME = 'alirezamsh/small100'
10 |
11 | model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12 |
13 | raw_dataset = load_dataset(DATASET, 'en-pl')
14 |
15 | tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 |                                           tgt_lang='pl')
17 |
18 | # The preprocessing function was adapted from the huggingface example
19 | # https://huggingface.co/docs/transformers/tasks/translation
20 | def tokenize_help(examples):
21 |     inputs = [ex['en'] for ex in examples['translation']]
22 |     targets = [ex['pl'] for ex in examples['translation']]
23 |     model_inputs = tokenizer(
24 |         inputs, text_target=targets, max_length=max_length, truncation=True
25 |     )
26 |     return model_inputs
27 |
28 |
29 | train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 |                  EPOCH_NUM, tokenize_help)
```

Training/TrainAlirezaEnPlPara.py

```
1 | from datasets import load_dataset
2 | from transformers import AutoTokenizer
3 |
4 | from training import train_model
5 |
6 | DATASET = 'para_crawl'
7 | EPOCH_NUM = 50
8 | max_length = 128
9 | MODEL_NAME = 'alirezamsh/small100'
10 |
11 | model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12 |
13 | raw_dataset = load_dataset(DATASET, 'enpl')
```

```

14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16     tgt_lang='pl')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en']] for ex in examples['translation']
22     targets = [ex['pl']] for ex in examples['translation']
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30     EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainAlirezaEnPlPara3.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14     0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17     tgt_lang='pl')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['en']] for ex in examples['translation']
23     targets = [ex['pl']] for ex in examples['translation']
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30     EPOCH_NUM, tokenize_help,
31     max_length)

```

Training/TrainAlirezaEnPlkde4.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'kde4'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='pl')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)
```

Training/TrainAlirezaPlEnCC.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'yhavinga/ccmatrix'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
```

```

15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16     tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30     EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainAlirezaPlEnOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16     tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30     EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainAlirezaPlEnOpusAcc.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_accelerate
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help)

```

Training/TrainAlirezaPlEnPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'enpl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example

```

```

19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainAlirezaPlEnPara3.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17     tgt_lang='en')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['pl'] for ex in examples['translation']]
23     targets = [ex['en'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30             EPOCH_NUM, tokenize_help,
31             max_length)

```

Training/TrainAlirezaPlEnkde4.py


```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'kde4'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'alirezamsh/small100'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnMulCC.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'yhavinga/ccmatrix'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation

```

```

20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnMulOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnMulOpusAcc.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_accelerate

```

```

5
6 DATASET = 'opus100'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help)

```

Training/TrainEnMulPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )

```

```

26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnMulPara3.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'para_crawl'
7  EPOCH_NUM = 100
8  max_length = 128
9  MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30             EPOCH_NUM, tokenize_help,
31             max_length)

```

Training/TrainEnMulkde4.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'kde4'
7  EPOCH_NUM = 100
8  max_length = 128

```

```

9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnPlCC.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'yhvinga/ccmatrix'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28

```

```

29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
EPOCH_NUM, tokenize_help, max_length)
30

```

Training/TrainEnPlOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
EPOCH_NUM, tokenize_help, max_length)
30

```

Training/TrainEnPlOpusAcc.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_accelerate
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')

```

```

14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help)

```

Training/TrainEnPlPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainEnPlPara3.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
0, with_indices=True)
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
EPOCH_NUM, tokenize_help,
30             max_length)
31

```

Training/TrainEnPlkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'kde4'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example

```



```

19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en']] for ex in examples['translation']
22     targets = [ex['pl']] for ex in examples['translation']
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainMulEnCC.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'yhavinga/ccmatrix'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl']] for ex in examples['translation']
22     targets = [ex['en']] for ex in examples['translation']
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainMulEnOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer

```

```

3
4 from training import train_model
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainMulEnOpusAcc.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_accelerate
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(

```

```

24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30                  EPOCH_NUM, tokenize_help)

```

Training/TrainMulEnPara.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'para_crawl'
7  EPOCH_NUM = 100
8  max_length = 128
9  MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30            EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainMulEnPara3.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'para_crawl'
7  EPOCH_NUM = 100
8  max_length = 128

```

```

9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30 EPOCH_NUM, tokenize_help,
31 max_length)

```

Training/TrainMulEnkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'kde4'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs

```

```

27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
EPOCH_NUM, tokenize_help, max_length)
30

```

Training/TrainNLLBEnPlCC.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'yhavinga/ccmatrix'
7  EPOCH_NUM = 50
8  max_length = 128
9  MODEL_NAME = 'facebook/nllb-200-distilled-600M'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
== 0, with_indices=True)
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
tgt_lang='pl')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
EPOCH_NUM, tokenize_help, max_length)
30

```

Training/TrainNLLBEnPlOpus.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'opus100'
7  EPOCH_NUM = 100
8  max_length = 128
9  MODEL_NAME = 'facebook/nllb-200-distilled-600M'

```

```

10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16 tgt_lang='pl')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['en'] for ex in examples['translation']]
22     targets = [ex['pl'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30 EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainNLLBEnPlPara3.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'facebook/nllb-200-distilled-600M'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-en-pl'
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17 tgt_lang='pl')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['en'] for ex in examples['translation']]
23     targets = [ex['pl'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28

```

```

29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30             EPOCH_NUM, tokenize_help,
31             max_length)

```

Training/TrainNLLBPlEnCC.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'yhavinga/ccmatrix'
7  EPOCH_NUM = 50
8  max_length = 128
9  MODEL_NAME = 'facebook/nllb-200-distilled-600M'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70
14 == 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17                                           tgt_lang='en')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['pl'] for ex in examples['translation']]
23     targets = [ex['en'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainNLLBPlEnOpus.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3
4  from training import train_model
5
6  DATASET = 'opus100'
7  EPOCH_NUM = 100
8  max_length = 128
9  MODEL_NAME = 'facebook/nllb-200-distilled-600M'
10

```

```

11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
16     tgt_lang='en')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30     EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainNLLBPlEnPara3.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'facebook/nllb-200-distilled-600M'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-') + '-pl-en'
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14     0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
17     tgt_lang='en')
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['pl'] for ex in examples['translation']]
23     targets = [ex['en'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',

```



```
30 EPOCH_NUM, tokenize_help,  
31     max_length)
```

Training/TrainPlEnCC.py

```
1 from datasets import load_dataset  
2 from transformers import AutoTokenizer  
3  
4 from training import train_model  
5  
6 DATASET = 'yhavinga/ccmatrix'  
7 EPOCH_NUM = 100  
8 max_length = 128  
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'  
10  
11 model_name_cleaned = MODEL_NAME.replace('/', '-')  
12  
13 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 70  
14 == 0, with_indices=True)  
15  
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')  
17  
18 # The preprocessing function was adapted from the huggingface example  
19 # https://huggingface.co/docs/transformers/tasks/translation  
20 def tokenize_help(examples):  
21     inputs = [ex['pl'] for ex in examples['translation']]  
22     targets = [ex['en'] for ex in examples['translation']]  
23     model_inputs = tokenizer(  
24         inputs, text_target=targets, max_length=max_length, truncation=True  
25     )  
26     return model_inputs  
27  
28  
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,  
30 EPOCH_NUM, tokenize_help, max_length)
```

Training/TrainPlEnOpus.py

```
1 from datasets import load_dataset  
2 from transformers import AutoTokenizer  
3  
4 from training import train_model  
5  
6 DATASET = 'opus100'  
7 EPOCH_NUM = 100  
8 max_length = 128  
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'  
10  
11 model_name_cleaned = MODEL_NAME.replace('/', '-')  
12
```

```

13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)

```

Training/TrainPlEnOpusAcc.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_accelerate
5
6 DATASET = 'opus100'
7 EPOCH_NUM = 50
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'en-pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_accelerate(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30                 EPOCH_NUM, tokenize_help)

```

Training/TrainPlEnPara.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)
```

Training/TrainPlEnPara3.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'para_crawl'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 3 ==
14 0, with_indices=True)
15
16 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
```

```

17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl']] for ex in examples['translation']]
22     targets = [ex['en']] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET + '-3',
30             EPOCH_NUM, tokenize_help,
31             max_length)

```

Training/TrainPlEnkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3
4 from training import train_model
5
6 DATASET = 'kde4'
7 EPOCH_NUM = 100
8 max_length = 128
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 model_name_cleaned = MODEL_NAME.replace('/', '-')
12
13 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
14
15 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl']] for ex in examples['translation']]
22     targets = [ex['en']] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 train_model(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned, DATASET,
30             EPOCH_NUM, tokenize_help, max_length)
31

```

Training/training.py

```

1 import torch
2 from accelerate import Accelerator
3 from huggingface_hub import login, get_full_repo_name, Repository
4 from torch.utils.data.dataloader import DataLoader
5 from tqdm import tqdm
6 from transformers import AutoModelForSeq2SeqLM, DataCollatorForSeq2Seq,
Seq2SeqTrainingArguments, Seq2SeqTrainer, AdamW, \
7     get_scheduler
8 import evaluate
9 import numpy as np
10
11
12 # the train function is adapted from the huggingface example
https://huggingface.co/docs/transformers/tasks/translation
13 def train_model(raw_dataset, tokenizer, model_name, model_name_cleaned,
dataset_name, epoch_num,
14                 tokenize_help, max_length, resume_from_checkpoint=False):
15     global metric
16     login('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')
17     split_datasets = raw_dataset['train'].train_test_split(train_size=0.9,
seed=20)
18     split_datasets['validation'] = split_datasets.pop('test')
19     tokenized_datasets = split_datasets.map(
20         tokenize_help,
21         batched=True,
22         remove_columns=split_datasets['train'].column_names,
23     )
24     model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
25     data_collator = DataCollatorForSeq2Seq(tokenizer, model=model)
26     metric = evaluate.load('sacrebleu')
27
28     dataset_name = dataset_name.replace('/', '-')
29
30     def compute_metrics(eval_preds):
31         preds, labels = eval_preds
32         # In case the model returns more than the prediction logits
33         if isinstance(preds, tuple):
34             preds = preds[0]
35
36         decoded_preds = tokenizer.batch_decode(preds, skip_special_tokens=True)
37
38         # Replace -100s in the labels as we can't decode them
39         labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
40         decoded_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)
41
42         # Some simple post-processing
43         decoded_preds = [pred.strip() for pred in decoded_preds]
44         decoded_labels = [[label.strip()] for label in decoded_labels]
45
46         result = metric.compute(predictions=decoded_preds,
references=decoded_labels)
47         return {'bleu': result['score']}
48
49     args = Seq2SeqTrainingArguments(
50         f'{model_name_cleaned}-{dataset_name}-finetune',
51         evaluation_strategy='no',
52         save_strategy='epoch',
53         learning_rate=2e-5,

```

```

54         per_device_train_batch_size=32,
55         per_device_eval_batch_size=64,
56         weight_decay=0.01,
57         save_total_limit=2,
58         num_train_epochs=epoch_num,
59         predict_with_generate=True,
60         fp16=True,
61         push_to_hub=True,
62     )
63     trainer = Seq2SeqTrainer(
64         model,
65         args,
66         train_dataset=tokenized_datasets['train'],
67         eval_dataset=tokenized_datasets['validation'],
68         data_collator=data_collator,
69         tokenizer=tokenizer,
70         compute_metrics=compute_metrics,
71     )
72     before_training = trainer.evaluate(max_length=max_length)
73     file = open(dataset_name + '-' + str(epoch_num) + '-' + model_name_cleaned +
74               '.txt', 'w')
75     file.write(str(before_training))
76     file.write('\n')
77     file.close()
78     trainer.train(resume_from_checkpoint=resume_from_checkpoint)
79     trainer.push_to_hub()
80
81     after_training = trainer.evaluate(max_length=max_length)
82     file = open(dataset_name + '-' + str(epoch_num) + '-' + model_name_cleaned +
83               '.txt', 'a')
84     file.write(str(after_training))
85     file.close()
86
87     # the accelerate function is adapted from the huggingface example
88     # https://huggingface.co/docs/accelerate/index
89     def train_accelerate(raw_dataset, tokenizer, model_name, model_name_cleaned,
90                          dataset_name, epoch_num,
91                          tokenize_help):
92         login('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')
93         split_datasets = raw_dataset['train'].train_test_split(train_size=0.9,
94 seed=20)
95         split_datasets['validation'] = split_datasets.pop('test')
96
97         tokenized_datasets = split_datasets.map(
98             tokenize_help,
99             batched=True,
100             remove_columns=split_datasets['train'].column_names,
101         )
102
103         model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
104
105         data_collator = DataCollatorForSeq2Seq(tokenizer, model=model)
106
107         metric = evaluate.load('sacrebleu')

```

```

108
109 dataset_name = dataset_name.replace('/', '-')
110
111 def compute_metrics(eval_preds):
112     preds, labels = eval_preds
113     # In case the model returns more than the prediction logits
114     if isinstance(preds, tuple):
115         preds = preds[0]
116
117     decoded_preds = tokenizer.batch_decode(preds, skip_special_tokens=True)
118
119     # Replace -100s in the labels as we can't decode them
120     labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
121     decoded_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)
122
123     # Some simple post-processing
124     decoded_preds = [pred.strip() for pred in decoded_preds]
125     decoded_labels = [[label.strip()] for label in decoded_labels]
126
127     result = metric.compute(predictions=decoded_preds,
128                             references=decoded_labels)
129     return {'bleu': result['score']}
130
131 tokenized_datasets.set_format('torch')
132 train_dataloader = DataLoader(
133     tokenized_datasets['train'],
134     shuffle=True,
135     collate_fn=data_collator,
136     batch_size=8,
137 )
138 eval_dataloader = DataLoader(
139     tokenized_datasets['validation'], collate_fn=data_collator, batch_size=8
140 )
141
142 model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
143
144 optimizer = AdamW(model.parameters(), lr=2e-5)
145
146 accelerator = Accelerator()
147 model, optimizer, train_dataloader, eval_dataloader = accelerator.prepare(
148     model, optimizer, train_dataloader, eval_dataloader
149 )
150
151 num_update_steps_per_epoch = len(train_dataloader)
152 num_training_steps = epoch_num * num_update_steps_per_epoch
153
154 lr_scheduler = get_scheduler(
155     'linear',
156     optimizer=optimizer,
157     num_warmup_steps=0,
158     num_training_steps=num_training_steps,
159 )
160
161 model_name = f'{model_name_cleaned}-{dataset_name}-accelerate'
162 repo_name = get_full_repo_name(model_name)
163
164 output_dir = f'{model_name_cleaned}-{dataset_name}-accelerate'

```

```

164     repo = Repository(output_dir, clone_from=repo_name)
165
166     def postprocess(predictions, labels):
167         predictions = predictions.cpu().numpy()
168         labels = labels.cpu().numpy()
169
170         decoded_preds = tokenizer.batch_decode(predictions,
133         skip_special_tokens=True)
171
172         # Replace -100 in the labels as we can't decode them.
173         labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
174         decoded_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)
175
176         # Some simple post-processing
177         decoded_preds = [pred.strip() for pred in decoded_preds]
178         decoded_labels = [[label.strip()] for label in decoded_labels]
179         return decoded_preds, decoded_labels
180
181     progress_bar = tqdm(range(num_training_steps))
182
183     for epoch in range(epoch_num):
184         # Training
185         model.train()
186         for batch in train_dataloader:
187             outputs = model(**batch)
188             loss = outputs.loss
189             accelerator.backward(loss)
190
191             optimizer.step()
192             lr_scheduler.step()
193             optimizer.zero_grad()
194             progress_bar.update(1)
195
196         # Evaluation
197         model.eval()
198         for batch in tqdm(eval_dataloader):
199             with torch.no_grad():
200                 generated_tokens = accelerator.unwrap_model(model).generate(
201                     batch['input_ids'],
202                     attention_mask=batch['attention_mask'],
203                     max_length=128,
204                 )
205                 labels = batch['labels']
206
207                 # Necessary to pad predictions and labels for being gathered
208                 generated_tokens = accelerator.pad_across_processes(
209                     generated_tokens, dim=1, pad_index=tokenizer.pad_token_id
210                 )
211                 labels = accelerator.pad_across_processes(labels, dim=1,
133                 pad_index=-100)
212
213                 predictions_gathered = accelerator.gather(generated_tokens)
214                 labels_gathered = accelerator.gather(labels)
215
216                 decoded_preds, decoded_labels = postprocess(predictions_gathered,
133                 labels_gathered)
217                 metric.add_batch(predictions=decoded_preds, references=decoded_labels)
218

```



```

219         results = metric.compute()
220
221         print(f"epoch {epoch}, BLEU score: {results['score']:.2f}")
222         file = open(dataset_name + '-' + str(epoch_num) + '-' + model_name_cleaned
+ '.txt', 'a')
223         file.write(f"epoch {epoch}, BLEU score: {results['score']:.2f}")
224         file.write('\n')
225         file.close()
226
227         # Save and upload
228         accelerator.wait_for_everyone()
229         unwrapped_model = accelerator.unwrap_model(model)
230         unwrapped_model.save_pretrained(output_dir,
save_function=accelerator.save)
231         if accelerator.is_main_process:
232             tokenizer.save_pretrained(output_dir)
233             repo.push_to_hub(
234                 commit_message=f'Training in progress epoch {epoch}',
blocking=False
235             )
236

```

19 files

```
Hyper/hyperAlirezaEnPlOpus.py
Hyper/hyperAlirezaEnPlPara.py
Hyper/hyperAlirezaEnPlkde4.py
Hyper/hyperAlirezaPlEnOpus.py
Hyper/hyperAlirezaPlEnPara.py
Hyper/hyperAlirezaPlEnkde4.py
Hyper/hyperEnMulOpus.py
Hyper/hyperEnMulPara.py
Hyper/hyperEnMulKde4.py
Hyper/hyperEnPlOpus.py
Hyper/hyperEnPlPara.py
Hyper/hyperEnPlkde4.py
Hyper/hyperMulEnOpus.py
Hyper/hyperMulEnPara.py
Hyper/hyperMulEnkde4.py
Hyper/hyperPlEnOpus.py
Hyper/hyperPlEnPara.py
Hyper/hyperPlEnkde4.py
Hyper/hyperparamSearch.py
```

Hyper/hyperAlirezaEnPlOpus.py

```
1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3  from hyperparamSearch import hyperparameter_search
4
5  EPOCH_NUM = 50
6  max_length = 128
7
8  DATASET = 'opus100'
9  MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 20
12 == 0, with_indices=True)
13
14 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
15 tgt_lang='pl')
16
17 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'en-pl'
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['en'] for ex in examples['translation']]
23     targets = [ex['pl'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
30 DATASET, EPOCH_NUM, tokenize_help)
```

Hyper/hyperAlirezaEnPlPara.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'para_crawl'
9 MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 50 ==
12 0, with_indices=True)
13
14 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
15 tgt_lang='pl')
16
17 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'en-pl'
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['en'] for ex in examples['translation']]
23     targets = [ex['pl'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
30 DATASET, EPOCH_NUM, tokenize_help)
```

Hyper/hyperAlirezaEnPlkde4.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'kde4'
9 MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
14 tgt_lang='pl')
```

```

15 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'en-pl'
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en'] for ex in examples['translation']]
21     targets = [ex['pl'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29     DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperAlirezaPlEnOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'opus100'
9 MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, 'en-pl').filter(lambda example, idx: idx % 20
12     == 0, with_indices=True)
13
14 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
15     tgt_lang='en')
16
17 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'pl-en'
18
19 # The preprocessing function was adapted from the huggingface example
20 # https://huggingface.co/docs/transformers/tasks/translation
21 def tokenize_help(examples):
22     inputs = [ex['pl'] for ex in examples['translation']]
23     targets = [ex['en'] for ex in examples['translation']]
24     model_inputs = tokenizer(
25         inputs, text_target=targets, max_length=max_length, truncation=True
26     )
27     return model_inputs
28
29 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
30     DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperAlirezaPlEnPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'para_crawl'
9 MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, 'enpl').filter(lambda example, idx: idx % 50 ==
0, with_indices=True)
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
tgt_lang='en')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'pl-en'
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['pl'] for ex in examples['translation']]
21     targets = [ex['en'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
DATASET, EPOCH_NUM, tokenize_help)
29

```

Hyper/hyperAlirezaPlEnkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'kde4'
9 MODEL_NAME = 'alirezamsh/small100'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
tgt_lang='en')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-') + 'pl-en'
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):

```

```

20     inputs = [ex['pl'] for ex in examples['translation']]
21     targets = [ex['en'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperEnMulOpus.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3  from hyperparamSearch import hyperparameter_search
4
5  EPOCH_NUM = 50
6  max_length = 128
7
8  DATASET = 'para_crawl'
9  MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 raw_dataset = load_dataset(DATASET, 'en-pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en'] for ex in examples['translation']]
21     targets = [ex['pl'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperEnMulPara.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3  from hyperparamSearch import hyperparameter_search
4
5  EPOCH_NUM = 50
6  max_length = 128

```

```

7
8 DATASET = 'para_crawl'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 raw_dataset = load_dataset(DATASET, 'enpl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en'] for ex in examples['translation']]
21     targets = [ex['pl'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29     DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperEnMulkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'kde4'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-en-mul'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en'] for ex in examples['translation']]
21     targets = [ex['pl'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,

```

```
29 | DATASET, EPOCH_NUM, tokenize_help)
```

Hyper/hyperEnPlOpus.py

```
1 | from datasets import load_dataset
2 | from transformers import AutoTokenizer
3 | from hyperparamSearch import hyperparameter_search
4 |
5 | EPOCH_NUM = 50
6 | max_length = 128
7 |
8 | DATASET = 'opus100'
9 | MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10 |
11 | raw_dataset = load_dataset(DATASET, 'en-pl')
12 |
13 | tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14 |
15 | model_name_cleaned = MODEL_NAME.replace('/', '-')
16 |
17 | # The preprocessing function was adapted from the huggingface example
18 | # https://huggingface.co/docs/transformers/tasks/translation
19 | def tokenize_help(examples):
20 |     inputs = [ex['en'] for ex in examples['translation']]
21 |     targets = [ex['pl'] for ex in examples['translation']]
22 |     model_inputs = tokenizer(
23 |         inputs, text_target=targets, max_length=max_length, truncation=True
24 |     )
25 |     return model_inputs
26 |
27 |
28 | hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 | DATASET, EPOCH_NUM, tokenize_help)
```

Hyper/hyperEnPlPara.py

```
1 | from datasets import load_dataset
2 | from transformers import AutoTokenizer
3 | from hyperparamSearch import hyperparameter_search
4 |
5 | EPOCH_NUM = 50
6 | max_length = 128
7 |
8 | DATASET = 'para_crawl'
9 | MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10 |
11 | raw_dataset = load_dataset(DATASET, 'enpl')
12 |
13 | tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14 |
15 | model_name_cleaned = MODEL_NAME.replace('/', '-')
```



```

16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en']] for ex in examples['translation']]
21     targets = [ex['pl']] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29     DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperEnPlkde4.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'kde4'
9 MODEL_NAME = 'gsarti/opus-mt-tc-en-pl'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['en']] for ex in examples['translation']]
21     targets = [ex['pl']] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29     DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperMulEnOpus.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer

```

```

3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'opus100'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 raw_dataset = load_dataset(DATASET, 'en-pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28
29 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
30                       DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperMulEnPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'para_crawl'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 raw_dataset = load_dataset(DATASET, 'enpl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True

```

```

24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperMulEnkde4.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3  from hyperparamSearch import hyperparameter_search
4
5  EPOCH_NUM = 50
6  max_length = 128
7
8  DATASET = 'kde4'
9  MODEL_NAME = 'Helsinki-NLP/opus-mt-mul-en'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt')
14
15 model_name_cleaned = MODEL_NAME.replace('/', '-')
16
17 # The preprocessing function was adapted from the huggingface example
18 # https://huggingface.co/docs/transformers/tasks/translation
19 def tokenize_help(examples):
20     inputs = [ex['pl'] for ex in examples['translation']]
21     targets = [ex['en'] for ex in examples['translation']]
22     model_inputs = tokenizer(
23         inputs, text_target=targets, max_length=max_length, truncation=True
24     )
25     return model_inputs
26
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperPlEnOpus.py

```

1  from datasets import load_dataset
2  from transformers import AutoTokenizer
3  from hyperparamSearch import hyperparameter_search
4
5  EPOCH_NUM = 50
6  max_length = 128
7
8  DATASET = 'opus100'
9  MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10

```

```

11 raw_dataset = load_dataset(DATASET, 'en-pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
14 tgt_lang='en')
15
16 model_name_cleaned = MODEL_NAME.replace('/', '-')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperPlEnPara.py

```

1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'para_crawl'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 raw_dataset = load_dataset(DATASET, 'enpl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
14 tgt_lang='en')
15
16 model_name_cleaned = MODEL_NAME.replace('/', '-')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)

```

Hyper/hyperPlEnkde4.py

```
1 from datasets import load_dataset
2 from transformers import AutoTokenizer
3 from hyperparamSearch import hyperparameter_search
4
5 EPOCH_NUM = 50
6 max_length = 128
7
8 DATASET = 'kde4'
9 MODEL_NAME = 'Helsinki-NLP/opus-mt-pl-en'
10
11 raw_dataset = load_dataset(DATASET, lang1='en', lang2='pl')
12
13 tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, return_tensors='pt',
14 tgt_lang='en')
15
16 model_name_cleaned = MODEL_NAME.replace('/', '-')
17
18 # The preprocessing function was adapted from the huggingface example
19 # https://huggingface.co/docs/transformers/tasks/translation
20 def tokenize_help(examples):
21     inputs = [ex['pl'] for ex in examples['translation']]
22     targets = [ex['en'] for ex in examples['translation']]
23     model_inputs = tokenizer(
24         inputs, text_target=targets, max_length=max_length, truncation=True
25     )
26     return model_inputs
27
28 hyperparameter_search(raw_dataset, tokenizer, MODEL_NAME, model_name_cleaned,
29 DATASET, EPOCH_NUM, tokenize_help)
```

Hyper/hyperparamSearch.py

```
1 import evaluate
2 import numpy as np
3 from transformers import AutoModelForSeq2SeqLM, DataCollatorForSeq2Seq,
4 Seq2SeqTrainingArguments, Seq2SeqTrainer
5 from huggingface_hub import login
6
7 # the beginning of the search function is adapted from the huggingface example
8 # https://huggingface.co/docs/transformers/tasks/translation
9 def hyperparameter_search(raw_dataset, tokenizer, model_name, model_name_cleaned,
10 dataset_name, epoch_num,
11 tokenize_help):
12     split_dataset = raw_dataset['train'].train_test_split(train_size=0.9, seed=20)
13
14     split_dataset['validation'] = split_dataset.pop('test')
15
16     login('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')
```

```

17 def init_model(trial):
18     return AutoModelForSeq2SeqLM.from_pretrained(model_name)
19
20 global metric
21 tokenized_datasets = split_dataset.map(
22     tokenize_help,
23     batched=True,
24     remove_columns=split_dataset['train'].column_names,
25 )
26 model = AutoModelForSeq2SeqLM.from_pretrained(model_name)
27 data_collator = DataCollatorForSeq2Seq(tokenizer, model=model)
28 metric = evaluate.load('sacrebleu')
29
30 def compute_metrics(eval_preds):
31     preds, labels = eval_preds
32     # In case the model returns more than the prediction logits
33     if isinstance(preds, tuple):
34         preds = preds[0]
35
36     decoded_preds = tokenizer.batch_decode(preds, skip_special_tokens=True)
37
38     # Replace -100s in the labels as we can't decode them
39     labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
40     decoded_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)
41
42     # Some simple post-processing
43     decoded_preds = [pred.strip() for pred in decoded_preds]
44     decoded_labels = [[label.strip()] for label in decoded_labels]
45
46     result = metric.compute(predictions=decoded_preds,
47                             references=decoded_labels)
48     return {'bleu': result['score']}
49
50 args = Seq2SeqTrainingArguments(
51     f'{model_name_cleaned}-{dataset_name}-finetune',
52     evaluation_strategy='no',
53     save_strategy='epoch',
54     learning_rate=2e-5,
55     per_device_train_batch_size=32,
56     per_device_eval_batch_size=64,
57     weight_decay=0.01,
58     save_total_limit=3,
59     num_train_epochs=epoch_num,
60     predict_with_generate=True,
61     fp16=True,
62     push_to_hub=True,
63 )
64 trainer = Seq2SeqTrainer(
65     model=None,
66     args=args,
67     train_dataset=tokenized_datasets['train'],
68     eval_dataset=tokenized_datasets['validation'],
69     data_collator=data_collator,
70     tokenizer=tokenizer,
71     compute_metrics=compute_metrics,
72     model_init=init_model,

```

```
73     best_trial = trainer.hyperparameter_search(  
74         direction='maximize',  
75         backend='optuna',  
76         n_trials=10,  
77     )  
78     print(best_trial)  
79     # save to file  
80     with open(f'{model_name_cleaned}-{dataset_name}-hyper.txt', 'w') as f:  
81         f.write(str(best_trial))  
82
```

16 files

```
Testing/TestAllModelsFromHub.py
Testing/TestModelEfficiency.py
Testing/TestModelScoresFlores.py
Testing/TestModelScoresFlores101.py
Testing/TestModelScoresFloresDev.py
Testing/TestModelScoresFloresTest.py
Testing/TestModelScoresIVA.py
Testing/TestModelScoresMix.py
Testing/TestModelScoresOpus.py
Testing/TestModelScoresTED14.py
Testing/TestModelScoresTED15.py
Testing/TestModelScoresTED16.py
Testing/TestModelScoresTatoeba.py
Testing/TestModelScoresTatoebaDev.py
Testing/TestModelScoresWMT.py
Testing/testing.py
```

Testing/TestAllModelsFromHub.py

```
1  from transformers import pipeline
2
3  from testing import get_model_params
4
5  sentence_en_to_pl = '''Hello, it's me
6  I was wondering if after all these years you'd like to meet
7  To go over everything
8  They say that time's supposed to heal ya, but I ain't done much healing
9  Hello, can you hear me?
10 I'm in California dreaming about who we used to be
11 When we were younger and free
12 I've forgotten how it felt before the world fell at our feet
13 There's such a difference between us
14 And a million miles
15 Hello from the other side
16 I must've called a thousand times
17 To tell you I'm sorry for everything that I've done
18 But when I call, you never seem to be home'''.split('\n')
19
20 sentence_pl_to_en = '''Ktoś pogasił wszystkie światła
21 Świat się ugiął od zamieci
22 I pomimo wielkich chęci
23 Nie mam dobrych wieści
24 Chciałoby się uciec
25 Nie przed wszystkim się da
26 Idzie zima
27 Wiem, że nigdy nie jest łatwa
28 Niesie ciemny dzień przy sobie
29 Nieruchome myśli w głowie
30 Żywe są o Tobie
31 Byle śpiewać było o czym
32 Serce było wciąż gorące
33 Znowu krótsze będą noce
34 Zobaczymy słońce
```



```

35 Chciałoby się uciec
36 Nie przed wszystkim się da
37 Chciałoby się uciec
38 Nie przed wszystkim się da
39 Idzie zima
40 Idzie zima''.split('\n')
41
42
43 def translate(sentence, model_name, tokenizer_to_use, source_lang=None,
target_lang=None):
44     translator = pipeline('translation', model=model_name,
tokenizer=tokenizer_to_use)
45     print(model_name)
46     if source_lang is not None:
47         results = translator(sentence, src_lang=source_lang, tgt_lang=target_lang)
48     else:
49         results = translator(sentence)
50     for index, result in enumerate(results):
51         print(result['translation_text'], ' -> ', sentence[index])
52         file.write(f"{result['translation_text']} -> {sentence[index]}")
53
54
55 params = get_model_params()
56
57 # open file
58 with open('results-translation.txt', 'w') as file:
59     for param in params:
60         translate(param['model_name'], param['tokenizer_to_use'],
param['source_lang'],
61                 param['additional_model_data'])
62

```

Testing/TestModelEfficiency.py

```

1 import time
2 from transformers import pipeline
3 from testing import get_model_params
4
5
6 def time_model(model_name, tokenizer_to_use, source_lang=None,
additional_model_data=None):
7     second_lang = 'pl' if source_lang == 'en' else 'en'
8     if additional_model_data is not None:
9         pipeline_to_use = pipeline('translation', tokenizer=tokenizer_to_use,
model=model_name, src_lang=source_lang,
10                                     tgt_lang=second_lang)
11     else:
12         pipeline_to_use = pipeline('translation', tokenizer=tokenizer_to_use,
model=model_name)
13
14     # time the model for 10 characters, 100 characters, and 300 characters
15     times = ''
16     for c in [10, 100, 300]:
17         # take an average of 10 runs
18         runs = []
19         for j in range(10):

```

```

20         text = 'a' * c
21         start = time.time()
22         pipeline_to_use(text)
23         end = time.time()
24         runs.append(end - start)
25         times += f'{sum(runs) / len(runs):.2f}, '
26
27     print(f'{model_name}: {times}')
28     with open('results-efficiency.txt', 'a') as file:
29         file.write(f'{model_name}: {times}\n')
30
31
32 params = get_model_params()
33
34 for param in params:
35     time_model(param['model_name'], param['tokenizer_to_use'],
36               param['source_lang'], param['additional_model_data'])

```

Testing/TestModelScoresFlores.py

```

1 from datasets import load_dataset, concatenate_datasets
2 from testing import evaluate_models
3
4 polish = load_dataset('facebook/flores', 'pol_Latn')
5 polish_dev = polish['dev'].select_columns(['sentence'])
6 polish_test = polish['devtest'].select_columns(['sentence'])
7 english = load_dataset('facebook/flores', 'eng_Latn')
8 english_dev = english['dev'].select_columns(['sentence']).rename_column('sentence',
9 'reference')
10 english_test = english['devtest'].select_columns(['sentence']).rename_column('sentence',
11 'reference')
12 combined_dataset_dev = concatenate_datasets([english_dev, polish_dev], axis=1)
13 combined_dataset_test = concatenate_datasets([english_test, polish_test], axis=1)
14 combined_dataset = concatenate_datasets([combined_dataset_dev,
15 combined_dataset_test], axis=0)
16
17 pl_en_dataset = combined_dataset.rename_column('sentence', 'source')
18
19 en_pl_dataset = combined_dataset.rename_column('reference', 'source')
20 en_pl_dataset = en_pl_dataset.rename_column('sentence', 'reference')
21
22 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-flores.txt')

```

Testing/TestModelScoresFlores101.py

```

1 from datasets import load_dataset, concatenate_datasets
2 from testing import evaluate_models
3
4 polish = load_dataset('gsarti/flores_101', 'pol')
5 polish_dev = polish['dev'].select_columns(['sentence'])
6 polish_test = polish['devtest'].select_columns(['sentence'])

```

```

7 english = load_dataset('gsarti/flores_101', 'eng')
8 english_dev = english['dev'].select_columns(['sentence']).rename_column('sentence',
'reference')
9 english_test =
english['devtest'].select_columns(['sentence']).rename_column('sentence',
'reference')
10 combined_dataset_dev = concatenate_datasets([english_dev, polish_dev], axis=1)
11 combined_dataset_test = concatenate_datasets([english_test, polish_test], axis=1)
12 combined_dataset = concatenate_datasets([combined_dataset_dev,
combined_dataset_test], axis=0)
13
14 pl_en_dataset = combined_dataset.rename_column('sentence', 'source')
15
16 en_pl_dataset = combined_dataset.rename_column('reference', 'source')
17 en_pl_dataset = en_pl_dataset.rename_column('sentence', 'reference')
18
19 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-flores101.txt')
20

```

Testing/TestModelScoresFloresDev.py

```

1 from datasets import load_dataset, concatenate_datasets
2 from testing import evaluate_models
3
4 polish = load_dataset('facebook/flores', 'pol_Latn')
5 polish_dev = polish['dev'].select_columns(['sentence'])
6 english = load_dataset('facebook/flores', 'eng_Latn')
7 english_dev = english['dev'].select_columns(['sentence']).rename_column('sentence',
'reference')
8 combined_dataset_dev = concatenate_datasets([english_dev, polish_dev], axis=1)
9
10 pl_en_dataset_dev = combined_dataset_dev.rename_column('sentence', 'source')
11
12 en_pl_dataset_dev = combined_dataset_dev.rename_column('reference', 'source')
13 en_pl_dataset_dev = en_pl_dataset_dev.rename_column('sentence', 'reference')
14
15 evaluate_models(en_pl_dataset_dev, pl_en_dataset_dev, 'results-flores-dev.txt')
16

```

Testing/TestModelScoresFloresTest.py

```

1 from datasets import load_dataset, concatenate_datasets
2 from testing import evaluate_models
3
4 polish = load_dataset('facebook/flores', 'pol_Latn')
5 polish_test = polish['devtest'].select_columns(['sentence'])
6 english = load_dataset('facebook/flores', 'eng_Latn')
7 english_test =
english['devtest'].select_columns(['sentence']).rename_column('sentence',
'reference')
8 combined_dataset_test = concatenate_datasets([english_test, polish_test], axis=1)
9
10 pl_en_dataset_test = combined_dataset_test.rename_column('sentence', 'source')
11

```

```

12 en_pl_dataset_test = combined_dataset_test.rename_column('reference', 'source')
13 en_pl_dataset_test = en_pl_dataset_test.rename_column('sentence', 'reference')
14
15 evaluate_models(en_pl_dataset_test, pl_en_dataset_test, 'results-flores-test.txt')
16

```

Testing/TestModelScoresIVA.py

```

1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('cartesinus/iva_mt_ws10t', split='test').flatten()
5 raw_dataset = raw_dataset.select_columns(['translation_utt.en',
6     'translation_utt.pl'])
7
8 en_pl_dataset = raw_dataset.rename_column('translation_utt.en', 'source')
9 en_pl_dataset = en_pl_dataset.rename_column('translation_utt.pl', 'reference')
10
11 pl_en_dataset = raw_dataset.rename_column('translation_utt.pl', 'source')
12 pl_en_dataset = pl_en_dataset.rename_column('translation_utt.en', 'reference')
13
14 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-iva.txt')
15

```

Testing/TestModelScoresMix.py

```

1 from datasets import load_dataset, concatenate_datasets, Dataset
2 from testing import evaluate_models
3
4 kde4 = load_dataset('kde4', lang1='en', lang2='pl',
5     split='train').select(range(100))
6 kde4 = kde4.flatten().select_columns(['translation.en', 'translation.pl'])
7 para_crawl = load_dataset('para_crawl', 'enpl', split='train').select(range(1300))
8 para_crawl = para_crawl.flatten().select_columns(['translation.en',
9     'translation.pl'])
10 opus100 = load_dataset('opus100', 'en-pl', split='test').select(range(1300))
11 opus100 = opus100.flatten().select_columns(['translation.en', 'translation.pl'])
12 ccmatrix = Dataset.from_list(list(load_dataset('yhvinga/ccmatrix', 'en-pl',
13     split='train', streaming=True).take(1300)))
14 ccmatrix = ccmatrix.flatten().select_columns(['translation.en', 'translation.pl'])
15
16 raw_dataset = concatenate_datasets([kde4, para_crawl, opus100, ccmatrix])
17
18 en_pl_dataset = raw_dataset.rename_column('translation.en', 'source')
19 en_pl_dataset = en_pl_dataset.rename_column('translation.pl', 'reference')
20
21 pl_en_dataset = raw_dataset.rename_column('translation.pl', 'source')
22 pl_en_dataset = pl_en_dataset.rename_column('translation.en', 'reference')
23
24 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-mix.txt')
25

```

Testing/TestModelScoresOpus.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('opus_euconst', 'en-pl', split='train').flatten()
5
6 en_pl_dataset = raw_dataset.rename_column('translation.en', 'source')
7 en_pl_dataset = en_pl_dataset.rename_column('translation.pl', 'reference')
8
9 pl_en_dataset = raw_dataset.rename_column('translation.pl', 'source')
10 pl_en_dataset = pl_en_dataset.rename_column('translation.en', 'reference')
11
12 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-euconst.txt')
13
```

Testing/TestModelScoresTED14.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('ted_talks_iwslt', language_pair=('en', 'pl'),
5 year='2014', split='test').flatten()
6
7 en_pl_dataset = raw_dataset.rename_column('translation.en', 'source')
8 en_pl_dataset = en_pl_dataset.rename_column('translation.pl', 'reference')
9
10 pl_en_dataset = raw_dataset.rename_column('translation.pl', 'source')
11 pl_en_dataset = pl_en_dataset.rename_column('translation.en', 'reference')
12
13 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-ted-14.txt')
```

Testing/TestModelScoresTED15.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('ted_talks_iwslt', language_pair=('en', 'pl'),
5 year='2015', split='test').flatten()
6
7 en_pl_dataset = raw_dataset.rename_column('translation.en', 'source')
8 en_pl_dataset = en_pl_dataset.rename_column('translation.pl', 'reference')
9
10 pl_en_dataset = raw_dataset.rename_column('translation.pl', 'source')
11 pl_en_dataset = pl_en_dataset.rename_column('translation.en', 'reference')
12
13 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-ted-15.txt')
```

Testing/TestModelScoresTED16.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('ted_talks_iwslt', language_pair=('en', 'pl'),
5                             year='2016', split='test').flatten()
6
7 en_pl_dataset = raw_dataset.rename_column('translation.en', 'source')
8 en_pl_dataset = en_pl_dataset.rename_column('translation.pl', 'reference')
9
10 pl_en_dataset = raw_dataset.rename_column('translation.pl', 'source')
11 pl_en_dataset = pl_en_dataset.rename_column('translation.en', 'reference')
12
13 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-ted-16.txt')
```

Testing/TestModelScoresTatoeba.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('Helsinki-NLP/tatoeba_mt', 'eng-pol',
5                             split='test').select(range(4000))
6
7 pl_en_dataset = raw_dataset.rename_column('sourceString', 'reference')
8 pl_en_dataset = pl_en_dataset.rename_column('targetString', 'source')
9
10 en_pl_dataset = raw_dataset.rename_column('targetString', 'reference')
11 en_pl_dataset = en_pl_dataset.rename_column('sourceString', 'source')
12
13 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-tatoeba.txt')
```

Testing/TestModelScoresTatoebaDev.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 raw_dataset = load_dataset('Helsinki-NLP/tatoeba_mt', 'eng-pol',
5                             split='dev').select(range(4000))
6
7 pl_en_dataset = raw_dataset.rename_column('sourceString', 'reference')
8 pl_en_dataset = pl_en_dataset.rename_column('targetString', 'source')
9
10 en_pl_dataset = raw_dataset.rename_column('targetString', 'reference')
11 en_pl_dataset = en_pl_dataset.rename_column('sourceString', 'source')
12
13 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-tatoeba-dev.txt')
```

Testing/TestModelScoresWMT.py

```
1 from datasets import load_dataset
2 from testing import evaluate_models
3
4 en_pl_dataset = load_dataset('gsarti/wmt_vat', 'wmt20_en_pl', split='test')
5 pl_en_dataset = load_dataset('gsarti/wmt_vat', 'wmt20_pl_en', split='test')
6
7 evaluate_models(en_pl_dataset, pl_en_dataset, 'results-wmt.txt')
8
```

Testing/testing.py

```
1 import evaluate
2 from transformers import pipeline
3
4
5 def evaluate_models(dataset_en_pl, dataset_pl_en, filename):
6     model_params = get_model_params()
7     for param in model_params:
8         evaluate_model(param['model_name'], param['tokenizer_to_use'], filename,
9                        dataset_pl_en, dataset_en_pl,
10                        param['source_lang'], param['additional_model_data'])
11
12 def evaluate_model(model_name, tokenizer_to_use, filename, pl_en_dataset,
13                  en_pl_dataset, source_lang=None,
14                  additional_model_data=None):
15     second_lang = 'pl' if source_lang == 'en' else 'en'
16     if source_lang == 'pl':
17         dataset = pl_en_dataset
18     else:
19         dataset = en_pl_dataset
20     if additional_model_data is not None:
21         pipeline_to_use = pipeline('translation', tokenizer=tokenizer_to_use,
22                                   model=model_name, src_lang=source_lang,
23                                   tgt_lang=second_lang)
24     else:
25         pipeline_to_use = pipeline('translation', tokenizer=tokenizer_to_use,
26                                   model=model_name)
27
28     task_evaluator = evaluate.evaluator('translation')
29
30     with open(filename, 'a') as file:
31         file.write('Evaluating model: ' + model_name + '\n')
32     print('Evaluating model: ' + model_name)
33
34     results = get_results(task_evaluator, pipeline_to_use, dataset)
35
36     with open(filename, 'a') as file:
37         file.write('Results ' + str(results['score']) + '\n')
38     print('Results ' + str(results['score']))
39
```

```

38 def get_results(task_evaluator_to_use, pipeline_to_use, dataset_to_use):
39     results = task_evaluator_to_use.compute(
40         model_or_pipeline=pipeline_to_use,
41         data=dataset_to_use,
42         input_column='source',
43         label_column='reference',
44         metric='sacrebleu'
45     )
46
47     return results
48
49
50 def get_model_params():
51     return [
52         {'model_name': 'gsarti/opus-mt-tc-en-pl', 'tokenizer_to_use':
53 'gsarti/opus-mt-tc-en-pl', 'source_lang': 'en',
54         'additional_model_data': None},
55         {'model_name': 'MikolajDeja/gsarti-opus-mt-tc-en-pl-kde4-finetune',
56         'tokenizer_to_use': 'gsarti/opus-mt-tc-en-pl',
57         'source_lang': 'en', 'additional_model_data': None},
58         {'model_name': 'MikolajDeja/gsarti-opus-mt-tc-en-pl-opus100-finetune',
59         'tokenizer_to_use': 'gsarti/opus-mt-tc-en-pl',
60         'source_lang': 'en', 'additional_model_data': None},
61         {'model_name': 'MikolajDeja/gsarti-opus-mt-tc-en-pl-3-para_crawl-
62 finetune',
63         'tokenizer_to_use': 'gsarti/opus-mt-tc-en-pl', 'source_lang': 'en',
64         'additional_model_data': None},
65         {'model_name': 'MikolajDeja/gsarti-opus-mt-tc-en-pl-para_crawl-finetune',
66         'tokenizer_to_use': 'gsarti/opus-mt-tc-en-pl', 'source_lang': 'en',
67         'additional_model_data': None},
68         {'model_name': 'MikolajDeja/gsarti-opus-mt-tc-en-pl-yhavinga-ccmatrix-
69 finetune',
70         'tokenizer_to_use': 'gsarti/opus-mt-tc-en-pl', 'source_lang': 'en',
71         'additional_model_data': None},
72         {'model_name': 'Helsinki-NLP/opus-mt-pl-en', 'tokenizer_to_use':
73 'Helsinki-NLP/opus-mt-pl-en',
74         'source_lang': 'pl',
75         'additional_model_data': None},
76         {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-pl-en-kde4-finetune',
77         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-pl-en', 'source_lang': 'pl',
78         'additional_model_data': None},
79         {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-pl-en-opus100-finetune',
80         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-pl-en', 'source_lang': 'pl',
81         'additional_model_data': None},
82         {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-pl-en-3-para_crawl-
83 finetune',
84         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-pl-en', 'source_lang': 'pl',
85         'additional_model_data': None},
86         {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-pl-en-para_crawl-
87 finetune',
88         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-pl-en', 'source_lang': 'pl',
89         'additional_model_data': None},
90         {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-pl-en-yhavinga-ccmatrix-
91 finetune',
92         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-pl-en', 'source_lang': 'pl',
93         'additional_model_data': None},
94         {'model_name': 'Helsinki-NLP/opus-mt-en-mul',
95         'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',

```



```

88         'additional_model_data': None},
89     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-en-mul-kde4-finetune',
90      'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',
91      'additional_model_data': None},
92     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-en-mul-opus100-finetune',
93      'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',
94      'additional_model_data': None},
95     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-en-mul-3-para_crawl-
finetune',
96      'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',
97      'additional_model_data': None},
98     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-en-mul-para_crawl-
finetune',
99      'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',
100     'additional_model_data': None},
101     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-en-mul-yhavinga-ccmatrix-
finetune',
102     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-en-mul', 'source_lang': 'en',
103     'additional_model_data': None},
104
105     {'model_name': 'Helsinki-NLP/opus-mt-mul-en',
106     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
107     'additional_model_data': None},
108     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-mul-en-kde4-finetune',
109     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
110     'additional_model_data': None},
111     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-mul-en-opus100-finetune',
112     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
113     'additional_model_data': None},
114     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-mul-en-3-para_crawl-
finetune',
115     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
116     'additional_model_data': None},
117     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-mul-en-para_crawl-
finetune',
118     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
119     'additional_model_data': None},
120     {'model_name': 'MikolajDeja/Helsinki-NLP-opus-mt-mul-en-yhavinga-ccmatrix-
finetune',
121     'tokenizer_to_use': 'Helsinki-NLP/opus-mt-mul-en', 'source_lang': 'pl',
122     'additional_model_data': None},
123
124     {'model_name': 'alirezamsh/small100', 'tokenizer_to_use':
'alirezamsh/small100', 'source_lang': 'pl',
125     'additional_model_data': 'pl'},
126     {'model_name': 'MikolajDeja/alirezamsh-small100-pl-en-kde4-finetune',
'tokenizer_to_use': 'alirezamsh/small100',
127     'source_lang': 'pl', 'additional_model_data': 'pl'},
128     {'model_name': 'MikolajDeja/alirezamsh-small100-pl-en-opus100-finetune',
'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
129     'additional_model_data': 'pl'},
130     {'model_name': 'MikolajDeja/alirezamsh-small100-pl-en-3-para_crawl-
finetune',
131     'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
132     'additional_model_data': 'pl'},
133     {'model_name': 'MikolajDeja/alirezamsh-small100-pl-en-para_crawl-
finetune',
'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
134     'additional_model_data': 'pl'},

```

```

134         {'model_name': 'MikolajDeja/alirezamsh-small100-pl-en-yhavinga-ccmatrix-
finetune',
135         'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'pl',
'additional_model_data': 'pl'},
136
137         {'model_name': 'alirezamsh/small100', 'tokenizer_to_use':
'alirezamsh/small100', 'source_lang': 'en',
138         'additional_model_data': 'en'},
139         {'model_name': 'MikolajDeja/alirezamsh-small100-en-pl-kde4-finetune',
'tokenizer_to_use': 'alirezamsh/small100',
140         'source_lang': 'en', 'additional_model_data': 'en'},
141         {'model_name': 'MikolajDeja/alirezamsh-small100-en-pl-opus100-finetune',
142         'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'en',
'additional_model_data': 'en'},
143         {'model_name': 'MikolajDeja/alirezamsh-small100-en-pl-3-para_crawl-
finetune',
144         'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'en',
'additional_model_data': 'en'},
145         {'model_name': 'MikolajDeja/alirezamsh-small100-en-pl-para_crawl-
finetune',
146         'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'en',
'additional_model_data': 'en'},
147         {'model_name': 'MikolajDeja/alirezamsh-small100-en-pl-yhavinga-ccmatrix-
finetune',
148         'tokenizer_to_use': 'alirezamsh/small100', 'source_lang': 'en',
'additional_model_data': 'en'},
149
150         {'model_name': 'facebook/nllb-200-distilled-600M', 'tokenizer_to_use':
'facebook/nllb-200-distilled-600M',
151         'source_lang': 'pl', 'additional_model_data': 'pl'},
152         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-opus-
finetune',
153         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'pl', 'additional_model_data': 'pl'},
154         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-3-
para_crawl-finetune',
155         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'pl', 'additional_model_data': 'pl'},
156         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-pl-en-
yhavinga-ccmatrix-finetune',
157         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'pl', 'additional_model_data': 'pl'},
158
159         {'model_name': 'facebook/nllb-200-distilled-600M', 'tokenizer_to_use':
'facebook/nllb-200-distilled-600M',
160         'source_lang': 'en', 'additional_model_data': 'en'},
161         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-opus-
finetune',
162         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'en', 'additional_model_data': 'en'},
163         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-3-
para_crawl-finetune',
164         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'en', 'additional_model_data': 'en'},
165         {'model_name': 'MikolajDeja/facebook-nllb-200-distilled-600M-en-pl-
yhavinga-ccmatrix-finetune',
166         'tokenizer_to_use': 'facebook/nllb-200-distilled-600M', 'source_lang':
'en', 'additional_model_data': 'en'},
167     ]
168

```

4 files

HelperScripts/acc.sh
HelperScripts/hyper.sh
HelperScripts/test.sh
HelperScripts/train.sh

HelperScripts/acc.sh

```
1  #!/bin/bash -l
2
3  #SBATCH --job-name=translation-training
4  #SBATCH --partition=nmes_gpu
5  #SBATCH --gres=gpu
6  #SBATCH --signal=USR2
7  #SBATCH --time=48:00:00
8  #SBATCH --mem=10240
9
10 module load anaconda3/2021.05-gcc-9.4.0
11 ml test_switch_kcl
12 source test_switch
13 ml git-lfs
14
15
16 source /users/${USER}/.bashrc
17 source activate /scratch/users/${USER}/conda/torch-env
18
19 pip install accelerate
20 pip install tqdm
21
22 python scripts/TrainAlirezaEnPlkde4.py
23 python scripts/TrainAlirezaPlEnkde4.py
24 python scripts/TrainEnMulkde4.py
25 python scripts/TrainEnPlkde4.py
26 python scripts/TrainMulEnkde4.py
27 python scripts/TrainPlEnkde4.py
28
29 python scripts/TrainAlirezaEnPlOpusAcc.py
30 python scripts/TrainAlirezaPlEnOpusAcc.py
31 python scripts/TrainPlEnOpusAcc.py
32 python scripts/TrainEnPlOpusAcc.py
33 python scripts/TrainEnMulOpusAcc.py
34 python scripts/TrainMulEnOpusAcc.py
35
36 python scripts/TrainAlirezaEnPlParaCrawl.py
37 python scripts/TrainAlirezaPlEnParaCrawl.py
38 python scripts/TrainPlEnPara.py
39 python scripts/TrainEnPlPara.py
40 python scripts/TrainEnMulPara.py
41 python scripts/TrainMulEnPara.py
42
43 python scripts/TrainAlirezaEnPlParaCrawl3.py
44 python scripts/TrainAlirezaPlEnParaCrawl3.py
```

```
45 python scripts/TrainPlEnPara3.py
46 python scripts/TrainEnPlPara3.py
47 python scripts/TrainEnMulPara3.py
48 python scripts/TrainMulEnPara3.py
```

HelperScripts/hyper.sh

```
1  #!/bin/bash -l
2
3  #SBATCH --job-name=hyperparam
4  #SBATCH --partition=nmes_gpu
5  #SBATCH --gres=gpu
6  #SBATCH --signal=USR2
7  #SBATCH --time=48:00:00
8  #SBATCH --mem=40960
9
10 module load anaconda3/2021.05-gcc-9.4.0
11
12 source /users/${USER}/.bashrc
13 source activate /scratch/users/${USER}/conda/torch-env
14
15
16 python scripts/hyperEnPlkde4.py
17 python scripts/hyperEnPlopus.py
18 python scripts/hyperEnPlpara.py
19 python scripts/hyperPlEnkde4.py
20 python scripts/hyperPlEnOpus.py
21 python scripts/hyperPlEnPara.py
22
23 python scripts/hyperEnMulKde4.py
24 python scripts/hyperEnMulOpus.py
25 python scripts/hyperEnMulPara.py
26
27 python scripts/hyperMulEnkde4.py
28 python scripts/hyperMulEnOpus.py
29 python scripts/hyperMulEnPara.py
30
```

HelperScripts/test.sh

```
1  #!/bin/bash -l
2
3  #SBATCH --job-name=testing
4  #SBATCH --partition=nmes_gpu
5  #SBATCH --gres=gpu
6  #SBATCH --signal=USR2
7  #SBATCH --time=48:00:00
8  #SBATCH --mem=10240
9
10 module load anaconda3/2021.05-gcc-9.4.0
11
12 source /users/${USER}/.bashrc
```

```

13 source activate /scratch/users/${USER}/conda/torch-env
14
15 pip install scipy>1.17
16 pip install --upgrade datasets
17
18 export TRANSFORMERS_CACHE=/scratch/users/k20010020/cache/
19
20 python scripts/TestModelScoresWMT.py
21 python scripts/TestModelScoresOpus.py
22 python scripts/TestModelScoresTatoeba.py
23 python scripts/TestModelScoresMix.py
24 python scripts/TestModelEfficiency.py
25 python scripts/TestModelScoresFlores.py
26 python scripts/TestModelScoresFloresTwo.py
27 python scripts/TestModelScoresFlores101.py
28 python scripts/TestModelScoresFlores101Two.py
29 python scripts/TestModelScoresOpus100.py
30 python scripts/TestModelScoresIVA.py
31 python scripts/TestModelScoresTED14.py
32 python scripts/TestModelScoresTED15.py
33 python scripts/TestModelScoresTED16.py

```

HelperScripts/train.sh

```

1 #!/bin/bash -l
2
3 #SBATCH --job-name=training
4 #SBATCH --partition=nmes_gpu
5 #SBATCH --gres=gpu
6 #SBATCH --signal=USR2
7 #SBATCH --time=48:00:00
8 #SBATCH --mem=40960
9
10 module load anaconda3/2021.05-gcc-9.4.0
11 ml test_switch_kcl
12 source test_switch
13 ml git-lfs
14
15
16 source /users/${USER}/.bashrc
17 source activate /scratch/users/${USER}/conda/torch-env
18
19 python scripts/TrainNLLBEnPlPara.py
20 python scripts/TrainNLLBEnPlOpus.py
21 python scripts/TrainNLLBEnPlPara3.py
22 python scripts/TrainNLLBPlEnPara3.py
23 python scripts/TrainNLLBPlEnOpus.py
24 python scripts/TrainNLLBEnPlCC.py
25 python scripts/TrainNLLBPlEnCC.py
26
27
28 python scripts/TrainAlirezaEnPlCC.py
29 python scripts/TrainAlirezaPlEnCC.py
30 python scripts/TrainEnPlCC.py
31 python scripts/TrainPlEnCC.py

```

```
32 python scripts/TrainEnMulCC.py
33 python scripts/TrainMulEnCC.py
34
35 python scripts/TrainAlirezaEnPlkde4.py
36 python scripts/TrainAlirezaPlEnkde4.py
37 python scripts/TrainEnMulkde4.py
38 python scripts/TrainEnPlkde4.py
39 python scripts/TrainMulEnkde4.py
40 python scripts/TrainPlEnkde4.py
41
42 python scripts/TrainAlirezaEnPlOpus100.py
43 python scripts/TrainAlirezaPlEnOpus100.py
44 python scripts/TrainPlEnOpus100.py
45 python scripts/TrainEnPlOpus100.py
46 python scripts/TrainEnMulOpus100.py
47 python scripts/TrainMulEnOpus100.py
48
49 python scripts/TrainAlirezaEnPlParaCrawl.py
50 python scripts/TrainAlirezaPlEnParaCrawl.py
51 python scripts/TrainPlEnPara.py
52 python scripts/TrainEnPlPara.py
53 python scripts/TrainEnMulPara.py
54 python scripts/TrainMulEnPara.py
55
56 python scripts/TrainAlirezaEnPlParaCrawl3.py
57 python scripts/TrainAlirezaPlEnParaCrawl3.py
58 python scripts/TrainPlEnPara3.py
59 python scripts/TrainEnPlPara3.py
60 python scripts/TrainEnMulPara3.py
61 python scripts/TrainMulEnPara3.py
```

60 files

```
Translator/.github/workflows/django.yml
Translator/.gitignore
Translator/README.md
Translator/manage.py
Translator/requirements.txt
Translator/static/custom.css
Translator/static/favicon/browserconfig.xml
Translator/static/favicon/safari-pinned-tab.svg
Translator/static/favicon/site.webmanifest
Translator/translate/__init__.py
Translator/translate/__pycache__/views.cpython-39.pyc.140357439501584
Translator/translate/admin.py
Translator/translate/apps.py
Translator/translate/forms.py
Translator/translate/migrations/0001_initial.py
Translator/translate/migrations/0002_translation.py
Translator/translate/migrations/0003_alter_translation_user.py
Translator/translate/migrations/0004_alter_translation_user.py
Translator/translate/migrations/0005_translation_input_language.py
Translator/translate/migrations/0006_remove_translation_created_at_and_more.py
Translator/translate/migrations/0007_translation_created_at_translation_updated_at.py
Translator/translate/migrations/__init__.py
Translator/translate/models.py
Translator/translate/templates/base.html
Translator/translate/templates/base_content.html
Translator/translate/templates/home.html
Translator/translate/templates/login.html
Translator/translate/templates/partials/bootstrap_form.html
Translator/translate/templates/partials/messages.html
Translator/translate/templates/partials/navbar.html
Translator/translate/templates/password_change.html
Translator/translate/templates/profile.html
Translator/translate/templates/signup.html
Translator/translate/tests/__init__.py
Translator/translate/tests/fixtures/default_translation.json
Translator/translate/tests/fixtures/default_user.json
Translator/translate/tests/fixtures/other_translations.json
Translator/translate/tests/fixtures/other_users.json
Translator/translate/tests/forms/__init__.py
Translator/translate/tests/forms/test_log_in_form.py
Translator/translate/tests/forms/test_sign_up_form.py
Translator/translate/tests/forms/test_translator_form.py
Translator/translate/tests/helpers.py
Translator/translate/tests/models/__init__.py
Translator/translate/tests/models/test_translation_model.py
Translator/translate/tests/models/test_user_model.py
Translator/translate/tests/views/__init__.py
Translator/translate/tests/views/test_change_password_view.py
Translator/translate/tests/views/test_home_view.py
Translator/translate/tests/views/test_log_in_view.py
Translator/translate/tests/views/test_logout_view.py
Translator/translate/tests/views/test_profile_view.py
Translator/translate/tests/views/test_sign_up_view.py
Translator/translate/views.py
Translator/translator/__init__.py
Translator/translator/asgi.py
Translator/translator/settings.py
Translator/translator/urls.py
Translator/translator/wsgi.py
Translator/translator_backend.py
```

Translator/.github/workflows/django.yml

```
1 name: Django CI
2
3 on:
4   push:
5     branches: [ "master" ]
6   pull_request:
7     branches: [ "master" ]
8
9 jobs:
10   build:
11
12     runs-on: ubuntu-latest
13     strategy:
14       max-parallel: 4
15       matrix:
16         python-version: [3.8, 3.9]
17
```

```

18     steps:
19     - uses: actions/checkout@v3
20     - name: Set up Python ${ matrix.python-version }}
21       uses: actions/setup-python@v3
22       with:
23         python-version: ${ matrix.python-version }}
24     - name: Install Dependencies
25       run: |
26         python -m pip install --upgrade pip
27         pip install -r requirements.txt
28     - name: Run Tests
29       run: |
30         python manage.py test
31

```

Translator/.gitignore

```

1  # Created by https://www.toptal.com/developers/gitignore/api/macos,pycharm,jetbrains,django,python
2  # Edit at https://www.toptal.com/developers/gitignore?templates=macos,pycharm,jetbrains,django,python
3
4  ### Django ###
5  *.log
6  *.pot
7  *.pyc
8  __pycache__/
9  local_settings.py
10 db.sqlite3
11 db.sqlite3-journal
12 media
13
14 # If your build process includes running collectstatic, then you probably don't need or want to include staticfiles/
15 # in your Git repository. Update and uncomment the following line accordingly.
16 # <django-project-name>/staticfiles/
17
18 ### Django.Python Stack ###
19 # Byte-compiled / optimized / DLL files
20 *.py[cod]
21 *$py.class
22
23 # C extensions
24 *.so
25
26 # Distribution / packaging
27 .Python
28 build/
29 develop-eggs/
30 dist/
31 downloads/
32 eggs/
33 .eggs/
34 lib/
35 lib64/
36 parts/
37 sdist/
38 var/
39 wheels/
40 share/python-wheels/
41 *.egg-info/
42 .installed.cfg
43 *.egg
44 MANIFEST
45
46 # PyInstaller
47 # Usually these files are written by a python script from a template
48 # before PyInstaller builds the exe, so as to inject date/other infos into it.
49 *.manifest
50 *.spec
51
52 # Installer logs
53 pip-log.txt
54 pip-delete-thisdirectory.txt
55
56 # Unit test / coverage reports
57 htmlcov/
58 .tox/
59 .nox/
60 .coverage

```



```

61 | .coverage.*
62 | .cache
63 | nosetests.xml
64 | coverage.xml
65 | *.cover
66 | *.py,cover
67 | .hypothesis/
68 | .pytest_cache/
69 | cover/
70 |
71 | # Translations
72 | *.mo
73 |
74 | # Django stuff:
75 |
76 | # Flask stuff:
77 | instance/
78 | .webassets-cache
79 |
80 | # Scrappy stuff:
81 | .scrappy
82 |
83 | # Sphinx documentation
84 | docs/_build/
85 |
86 | # PyBuilder
87 | .pybuilder/
88 | target/
89 |
90 | # Jupyter Notebook
91 | .ipynb_checkpoints
92 |
93 | # IPython
94 | profile_default/
95 | ipython_config.py
96 |
97 | # pyenv
98 | #   For a library or package, you might want to ignore these files since the code is
99 | #   intended to run in multiple environments; otherwise, check them in:
100 | # .python-version
101 |
102 | # pipenv
103 | #   According to pya/pipenv#598, it is recommended to include Pipfile.lock in version control.
104 | #   However, in case of collaboration, if having platform-specific dependencies or dependencies
105 | #   having no cross-platform support, pipenv may install dependencies that don't work, or not
106 | #   install all needed dependencies.
107 | #Pipfile.lock
108 |
109 | # poetry
110 | #   Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
111 | #   This is especially recommended for binary packages to ensure reproducibility, and is more
112 | #   commonly ignored for libraries.
113 | #   https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
114 | #poetry.lock
115 |
116 | # pdm
117 | #   Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
118 | #pdm.lock
119 | #   pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
120 | #   in version control.
121 | #   https://pdm.fming.dev/#use-with-ide
122 | .pdm.toml
123 |
124 | # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
125 | __pypackages__/
126 |
127 | # Celery stuff
128 | celerybeat-schedule
129 | celerybeat.pid
130 |
131 | # SageMath parsed files
132 | *.sage.py
133 |
134 | # Environments
135 | .env
136 | .venv
137 | env/
138 | venv/
139 | ENV/

```

```
140 env.bak/
141 venv.bak/
142
143 # Spyder project settings
144 .spyderproject
145 .spyproject
146
147 # Rope project settings
148 .ropeproject
149
150 # mkdocs documentation
151 /site
152
153 # mypy
154 .mypy_cache/
155 .dmypy.json
156 dmypy.json
157
158 # Pyre type checker
159 .pyre/
160
161 # pytype static type analyzer
162 .pytype/
163
164 # Cython debug symbols
165 cython_debug/
166
167 # PyCharm
168 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
169 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
170 # and can be added to the global gitignore or merged into this file. For a more nuclear
171 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
172 #.idea/
173
174 ### JetBrains ###
175 # Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio, WebStorm and Rider
176 # Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839
177
178 # User-specific stuff
179 .idea/**/workspace.xml
180 .idea/**/tasks.xml
181 .idea/**/usage.statistics.xml
182 .idea/**/dictionaries
183 .idea/**/shelf
184
185 # AWS User-specific
186 .idea/**/aws.xml
187
188 # Generated files
189 .idea/**/contentModel.xml
190
191 # Sensitive or high-churn files
192 .idea/**/dataSources/
193 .idea/**/dataSources.ids
194 .idea/**/dataSources.local.xml
195 .idea/**/sqlDataSources.xml
196 .idea/**/dynamic.xml
197 .idea/**/uiDesigner.xml
198 .idea/**/dbnavigator.xml
199
200 # Gradle
201 .idea/**/gradle.xml
202 .idea/**/libraries
203
204 # Gradle and Maven with auto-import
205 # When using Gradle or Maven with auto-import, you should exclude module files,
206 # since they will be recreated, and may cause churn. Uncomment if using
207 # auto-import.
208 # .idea/artifacts
209 # .idea/compiler.xml
210 # .idea/jarRepositories.xml
211 # .idea/modules.xml
212 # .idea/*.iml
213 # .idea/modules
214 # *.iml
215 # *.ipr
216
217 # CMake
218 cmake-build-*/
```

```
219
220 # Mongo Explorer plugin
221 .idea/**/mongoSettings.xml
222
223 # File-based project format
224 *.iws
225
226 # IntelliJ
227 out/
228
229 # mpeltonen/sbt-idea plugin
230 .idea_modules/
231
232 # JIRA plugin
233 atlassian-ide-plugin.xml
234
235 # Cursive Clojure plugin
236 .idea/replstate.xml
237
238 # SonarLint plugin
239 .idea/sonarlint/
240
241 # Crashlytics plugin (for Android Studio and IntelliJ)
242 com_crashlytics_export_strings.xml
243 crashlytics.properties
244 crashlytics-build.properties
245 fabric.properties
246
247 # Editor-based Rest Client
248 .idea/httpRequests
249
250 # Android studio 3.1+ serialized cache file
251 .idea/caches/build_file_checksums.ser
252
253 ### JetBrains Patch ###
254 # Comment Reason: https://github.com/joeb lau/gitignore.io/issues/186#issuecomment-215987721
255
256 # *.iml
257 # modules.xml
258 # .idea/misc.xml
259 # *.ipr
260
261 # Sonarlint plugin
262 # https://plugins.jetbrains.com/plugin/7973-sonarlint
263 .idea/**/sonarlint/
264
265 # SonarQube Plugin
266 # https://plugins.jetbrains.com/plugin/7238-sonarqube-community-plugin
267 .idea/**/sonarIssues.xml
268
269 # Markdown Navigator plugin
270 # https://plugins.jetbrains.com/plugin/7896-markdown-navigator-enhanced
271 .idea/**/markdown-navigator.xml
272 .idea/**/markdown-navigator-enh.xml
273 .idea/**/markdown-navigator/
274
275 # Cache file creation bug
276 # See https://youtrack.jetbrains.com/issue/JBR-2257
277 .idea/$CACHE_FILE$
278
279 # CodeStream plugin
280 # https://plugins.jetbrains.com/plugin/12206-codestream
281 .idea/codestream.xml
282
283 # Azure Toolkit for IntelliJ plugin
284 # https://plugins.jetbrains.com/plugin/8053-azure-toolkit-for-intellij
285 .idea/**/azureSettings.xml
286
287 ### macOS ###
288 # General
289 .DS_Store
290 .AppleDouble
291 .LSOVERRIDE
292
293 # Icon must end with two \r
294 Icon
295
296
297 # Thumbnails
```

```
298 | ._*
299 |
300 | # Files that might appear in the root of a volume
301 | .DocumentRevisions-V100
302 | .fsevents
303 | .Spotlight-V100
304 | .TemporaryItems
305 | .Trashes
306 | .VolumeIcon.icns
307 | .com.apple.timemachine.donotpresent
308 |
309 | # Directories potentially created on remote AFP share
310 | .AppleDB
311 | .AppleDesktop
312 | Network Trash Folder
313 | Temporary Items
314 | .apdisk
315 |
316 | ### macOS Patch ###
317 | # iCloud generated files
318 | *.icloud
319 |
320 | ### PyCharm ###
321 | # Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio, WebStorm and Rider
322 | # Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839
323 |
324 | # User-specific stuff
325 |
326 | # AWS User-specific
327 |
328 | # Generated files
329 |
330 | # Sensitive or high-churn files
331 |
332 | # Gradle
333 |
334 | # Gradle and Maven with auto-import
335 | # When using Gradle or Maven with auto-import, you should exclude module files,
336 | # since they will be recreated, and may cause churn. Uncomment if using
337 | # auto-import.
338 | # .idea/artifacts
339 | # .idea/compiler.xml
340 | # .idea/jarRepositories.xml
341 | # .idea/modules.xml
342 | # .idea/*.iml
343 | # .idea/modules
344 | # *.iml
345 | # *.ipr
346 |
347 | # CMake
348 |
349 | # Mongo Explorer plugin
350 |
351 | # File-based project format
352 |
353 | # IntelliJ
354 |
355 | # mpeltonen/sbt-idea plugin
356 |
357 | # JIRA plugin
358 |
359 | # Cursive Clojure plugin
360 |
361 | # SonarLint plugin
362 |
363 | # Crashlytics plugin (for Android Studio and IntelliJ)
364 |
365 | # Editor-based Rest Client
366 |
367 | # Android studio 3.1+ serialized cache file
368 |
369 | ### PyCharm Patch ###
370 | # Comment Reason: https://github.com/joeblau/gitignore.io/issues/186#issuecomment-215987721
371 |
372 | # *.iml
373 | # modules.xml
374 | # .idea/misc.xml
375 | # *.ipr
376 |
```

```
377 # Sonarlint plugin
378 # https://plugins.jetbrains.com/plugin/7973-sonarlint
379
380 # SonarQube Plugin
381 # https://plugins.jetbrains.com/plugin/7238-sonarqube-community-plugin
382
383 # Markdown Navigator plugin
384 # https://plugins.jetbrains.com/plugin/7896-markdown-navigator-enhanced
385
386 # Cache file creation bug
387 # See https://youtrack.jetbrains.com/issue/JBR-2257
388
389 # CodeStream plugin
390 # https://plugins.jetbrains.com/plugin/12206-codestream
391
392 # Azure Toolkit for IntelliJ plugin
393 # https://plugins.jetbrains.com/plugin/8053-azure-toolkit-for-intellij
394
395 ### Python ###
396 # Byte-compiled / optimized / DLL files
397
398 # C extensions
399
400 # Distribution / packaging
401
402 # PyInstaller
403 # Usually these files are written by a python script from a template
404 # before PyInstaller builds the exe, so as to inject date/other infos into it.
405
406 # Installer logs
407
408 # Unit test / coverage reports
409
410 # Translations
411
412 # Django stuff:
413
414 # Flask stuff:
415
416 # Scrapy stuff:
417
418 # Sphinx documentation
419
420 # PyBuilder
421
422 # Jupyter Notebook
423
424 # IPython
425
426 # pyenv
427 # For a library or package, you might want to ignore these files since the code is
428 # intended to run in multiple environments; otherwise, check them in:
429 # .python-version
430
431 # pipenv
432 # According to pypa/pipenv#598, it is recommended to include Pipfile.lock in version control.
433 # However, in case of collaboration, if having platform-specific dependencies or dependencies
434 # having no cross-platform support, pipenv may install dependencies that don't work, or not
435 # install all needed dependencies.
436
437 # poetry
438 # Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
439 # This is especially recommended for binary packages to ensure reproducibility, and is more
440 # commonly ignored for libraries.
441 # https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
442
443 # pdm
444 # Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
445 # pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
446 # in version control.
447 # https://pdm.fming.dev/#use-with-ide
448
449 # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
450
451 # Celery stuff
452
453 # SageMath parsed files
454
455 # Environments
```

```

456
457 # Spyder project settings
458
459 # Rope project settings
460
461 # mkdocs documentation
462
463 # mypy
464
465 # Pyre type checker
466
467 # pytype static type analyzer
468
469 # Cython debug symbols
470
471 # PyCharm
472 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
473 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
474 # and can be added to the global gitignore or merged into this file. For a more nuclear
475 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
476
477 ### Python Patch ###
478 # Poetry local configuration file - https://python-poetry.org/docs/configuration/#local-configuration
479 poetry.toml
480
481 # ruff
482 .ruff_cache/
483
484 # End of https://www.toptal.com/developers/gitignore/api/macOS,pycharm,jetbrains,django,python

```

Translator/README.md

Translator

This is an app utilising the models I trained for my final year project.

It is a simple Django app, with simple user management. It allows translating between English and Polish, both ways. It also stores past translations for logged-in users.

Installation

To install the app, you need to have Python 3 installed (3.9 recommended). Then, you need to install the requirements:

```

virtualenv venv --python=3.9
source venv/bin/activate
pip3 install -r requirements.txt

```

Then, you need to create a database:

```
python3 manage.py migrate
```

Then, you need to run the server:

```
python3 manage.py runserver
```

To run tests, use the following command:

```
python3 manage.py test
```

To run coverage, use the following command:

```

coverage run manage.py test
coverage report

```

Usage

To use the app, navigate to localhost:8000 in your browser. You can then register an account, and log in if you choose to do so. You can translate between English and Polish, both ways.

Translator/manage.py

```

1  #!/usr/bin/env python
2  """Django's command-line utility for administrative tasks."""
3  import os
4  import sys
5
6
7  def main():
8      """Run administrative tasks."""
9      os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'translator.settings')
10     try:
11         from django.core.management import execute_from_command_line
12     except ImportError as exc: # pragma: no cover
13         raise ImportError(
14             "Couldn't import Django. Are you sure it's installed and "
15             "available on your PYTHONPATH environment variable? Did you "
16             "forget to activate a virtual environment?"
17         ) from exc
18     execute_from_command_line(sys.argv)
19
20
21 if __name__ == '__main__':
22     main()
23

```

Translator/requirements.txt

```

1  Django==4.1.5
2  django-widget-tweaks==1.4.12
3  django-extensions==3.2.1
4  transformers==4.26.0
5  torch==1.13.1
6  sentencepiece==0.1.97
7  coverage==7.1.0
8  django-extensions==3.2.1
9

```

Translator/static/custom.css

```

1  .form-box {
2      background-color: #141414;
3      padding-bottom: 2%;
4      padding-top: 2%;
5      border-radius: 15px;
6  }
7
8  .form-button {
9      background-color: #6D1DBC;
10     border-color: #6D1DBC;
11     color: #FBF6FF;
12 }
13
14 .form-button:hover {
15     background-color: #5f1aa3;
16     border-color: #5f1aa3;
17     color: #FBF6FF;
18 }
19
20 .form-button:active {
21     background-color: #551791;
22     border-color: #551791;
23     color: #FBF6FF;
24 }
25
26 .translation-box {
27     background-color: #141414;
28     padding: 2%;
29     border-radius: 15px;
30 }
31
32 .translation-item {
33     background-color: #282828;
34     padding: 1%;
35     margin: 1%;
36 }
37

```

```

36     border-radius: 15px;
37     text-align: center;
38 }
39
40 .translation-item:hover {
41     background-color: ■ #3a3a3a;
42 }

```

Translator/static/favicon/browserconfig.xml

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <browserconfig>
3     <msapplication>
4         <tile>
5             <square150x150logo src="/mstile-150x150.png"/>
6             <TileColor>#fbf6ff</TileColor>
7         </tile>
8     </msapplication>
9 </browserconfig>
10

```

Translator/static/favicon/safari-pinned-tab.svg

```

1 <?xml version="1.0" standalone="no"?>
2 <!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20010904//EN"
3 "http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
4 <svg version="1.0" xmlns="http://www.w3.org/2000/svg"
5 width="640.000000pt" height="640.000000pt" viewBox="0 0 640.000000 640.000000"
6 preserveAspectRatio="xMidYMid meet">
7 <metadata>
8 Created by potrace 1.14, written by Peter Selinger 2001-2017
9 </metadata>
10 <g transform="translate(0.000000,640.000000) scale(0.100000,-0.100000)"
11 fill="#000000" stroke="none">
12 <path d="M604 5121 c-2 -2 -23 -5 -46 -7 -52 -5 -208 -54 -208 -65 0 -5 -6 -9
13 -12 -9 -24 -1 -103 -61 -166 -127 -64 -68 -136 -196 -147 -260 -3 -18 -9 -44
14 -15 -58 -6 -16 -10 -525 -9 -1383 0 -1496 -3 -1425 63 -1565 41 -87 52 -103
15 138 -188 44 -43 93 -79 144 -105 42 -21 81 -40 86 -40 4 -1 9 -2 11 -4 1 -1 9
16 -3 17 -5 8 -2 29 -6 45 -10 17 -4 72 -8 122 -9 51 0 93 -3 93 -6 0 -3 26 -4
17 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58
18 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28
19 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1
20 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0
21 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57
22 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2
23 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2
24 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62
25 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3
26 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3
27 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58
28 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63
29 3 34 1 62 0 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0
30 62 -3 0 -3 26 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -3 26
31 -4 57 -3 32 1 58 2 58 3 0 1 28 2 63 3 34 1 62 0 62 -3 0 -6 109 -4 116 2 2 3
32 24 2 49 -2 24 -4 50 -3 57 1 6 4 15 6 19 4 5 -4 67 -6 119 -4 8 0 33 0 55 0
33 22 0 47 0 55 0 8 0 40 0 70 0 30 0 64 -1 75 -1 11 0 36 0 55 0 63 0 167 7 177
34 13 5 3 25 9 44 12 66 11 193 82 262 147 66 63 126 142 127 166 0 6 4 12 8 12
35 5 0 22 37 38 82 129 83 2 1375 c3 1518 6 1442 -61 1583 -39 84 -52 101 -127
36 178 -46 46 -157 129 -174 129 -2 0 -16 5 -32 12 -15 6 -31 12 -35 13 -5 2 -9
37 3 -10 5 -2 1 -10 3 -18 5 -8 2 -28 6 -45 10 -32 7 -109 14 -160 15 -34 0 -49
38 0 -120 -1 -27 0 -57 0 -65 0 -8 1 -33 1 -55 0 -38 -1 -47 0 -77 2 -14 1 -112
39 0 -168 -2 -19 -1 -41 -1 -47 0 -7 1 -19 1 -25 2 -14 1 -112 0 -168 -2 -19 -1
40 -41 -1 -47 0 -7 1 -19 1 -25 2 -14 1 -112 0 -168 -2 -19 -1 -41 -1 -47 0 -7 1
41 -19 1 -25 2 -14 1 -112 0 -168 -2 -19 -1 -41 -1 -47 0 -7 1 -19 1 -25 2 -14 1
42 -112 0 -168 -2 -19 -1 -41 -1 -47 0 -7 1 -19 1 -25 2 -14 1 -112 0 -168 -2
43 -19 -1 -39 -1 -45 -1 -13 1 -395 1 -415 1 -8 -1 -80 0 -160 0 -80 0 -152 0
44 -160 0 -8 0 -80 0 -160 0 -80 0 -152 0 -160 0 -8 0 -80 0 -160 0 -178 1 -362
45 2 -375 2 -82 -1 -103 -1 -190 0 -33 0 -67 0 -75 0 -8 -1 -31 -1 -50 0 -19 0
46 -42 0 -50 0 -8 0 -40 0 -70 0 -30 0 -62 0 -70 0 -8 -1 -31 -1 -50 0 -19 0 -42
47 0 -50 0 -8 0 -40 0 -70 0 -30 0 -62 0 -70 0 -8 -1 -31 -1 -50 0 -19 0 -42 0
48 -50 0 -8 0 -40 0 -70 0 -30 0 -62 0 -70 0 -8 -1 -31 -1 -50 0 -19 1 -39 1 -45
49 0 -70 -4 -135 -5 -135 -1 0 3 -26 4 -57 3 -32 -1 -58 -2 -58 -3 0 -1 -28 -2
50 -63 -3 -34 -1 -62 0 -62 3 0 3 -26 4 -57 3 -32 -1 -58 -2 -58 -3 0 -1 -28 -2
51 -63 -3 -34 -1 -62 0 -62 3 0 6 -110 7 -116 1z m2818 -640 c6 4 15 6 19 4 6 -4
52 80 -6 119 -4 8 1 29 0 45 0 26 -1 130 -3 160 -3 6 0 71 1 145 1 74 0 144 0

```



```

53 155 0 11 0 85 0 165 0 80 0 151 0 158 0 19 0 181 0 195 0 6 -1 37 -1 67 0 30
54 0 62 0 70 0 8 -1 31 -1 50 0 19 1 42 1 50 0 8 -1 40 -1 70 0 30 0 62 0 70 0 8
55 -1 31 -1 50 0 19 1 42 1 50 0 8 -1 40 -1 70 0 30 0 62 0 70 0 8 -1 31 -1 50 0
56 19 1 42 1 50 0 8 -1 40 -1 70 0 30 0 62 0 70 0 8 -1 31 -1 50 0 19 1 42 1 50
57 0 8 -1 38 -1 65 0 153 3 154 3 154 -16 0 -10 0 -36 0 -58 0 -22 0 -47 0 -55 0
58 -8 0 -31 0 -50 1 -19 1 -42 0 -50 0 -8 0 -42 0 -75 1 -85 1 -131 0 -190 0 -5
59 0 -66 1 -135 0 -69 0 -127 0 -130 -4 -19 -4 -120 0 -120 3 0 4 -30 2 -67 -4
60 -90 -2 -165 4 -175 3 -4 0 -8 -6 -8 -6 0 -9 -4 -6 -9 5 -7 7 -82 5 -131 0 -8
61 0 -40 0 -70 0 -30 0 -62 0 -70 0 -8 0 -31 0 -50 1 -19 1 -42 0 -50 0 -8 0 -42
62 0 -75 1 -85 1 -131 0 -190 0 -5 0 -66 1 -135 0 -69 0 -127 0 -130 -4 -19 -4
63 -120 0 -120 3 0 4 -28 3 -62 -1 -35 -2 -63 -3 -63 -1 0 -2 -26 -3 -58 -1 -31
64 1 -57 4 -57 3 0 3 -20 -1 -45 -4 -24 -4 -49 0 -55 4 -6 5 -22 2 -36 -4 -20
65 -11 -24 -41 -25 -20 -1 -40 0 -46 1 -5 1 -32 1 -60 0 -27 -1 -57 -1 -65 -1 -8
66 1 -33 1 -55 0 -38 -1 -47 0 -77 2 -7 1 -38 1 -68 0 -30 0 -62 0 -70 0 -8 1
67 -31 1 -50 0 -19 -1 -42 -1 -50 0 -8 1 -40 1 -70 0 -30 0 -62 0 -70 0 -8 1 -31
68 1 -50 0 -19 -1 -42 -1 -50 0 -8 1 -40 1 -70 0 -30 0 -62 0 -70 0 -8 1 -31 1
69 -50 0 -19 -1 -42 -1 -50 0 -8 1 -40 1 -70 0 -30 0 -62 -1 -70 0 -14 0 -264 1
70 -305 0 -11 0 -85 0 -165 0 -80 0 -152 0 -160 0 -8 0 -62 0 -120 0 -58 0 -109
71 0 -115 0 -82 -1 -103 -1 -190 0 -73 1 -75 1 -100 -1 -11 -1 -38 -1 -60 -1 -22
72 0 -47 0 -55 0 -8 0 -32 1 -53 2 1 -38 2 1 46 c0 25 -1 53 -1 61 -1 8 -1 38 0
73 65 0 28 1 57 0 65 -1 8 -1 31 0 50 2 56 3 154 2 168 -1 6 -1 18 -2 25 -1 6 -1
74 28 0 47 2 56 3 154 2 168 -1 6 -1 18 -2 25 -1 6 -1 28 0 47 2 56 3 154 2 168
75 -1 6 -1 18 -2 25 -1 6 -1 28 0 47 2 48 3 152 2 168 0 13 0 175 0 195 0 6 0 77
76 0 157 0 80 0 152 0 160 0 8 0 80 0 160 0 80 0 152 0 160 0 8 0 80 0 160 0 80
77 0 152 0 160 0 8 0 59 -1 112 -1 54 2 103 5 109 4 6 29 9 56 8 27 -2 51 0 54 3
78 3 2 25 2 50 -2 24 -4 50 -3 57 1z m-1711 -396 c21 -15 44 -37 52 -49 8 -11 58
79 -120 112 -241 53 -121 107 -240 118 -265 11 -25 21 -47 22 -50 1 -3 8 -21 17
80 -40 8 -19 43 -96 76 -170 33 -74 71 -160 85 -190 13 -30 50 -113 81 -185 58
81 -131 67 -152 126 -280 51 -111 49 -185 -8 -262 -67 -91 -233 -92 -303 -1 -13
82 16 -45 78 -71 136 1-47 107 -138 3 c-169 3 -135 3 -391 2 1-213 -1 -9 -25 c-9
83 -23 -26 -61 -78 -174 -34 -76 -100 -117 -182 -115 -117 3 -188 70 -196 182 -3
84 57 0 68 95 276 17 37 31 69 31 72 0 2 15 36 34 77 18 40 56 125 84 188 27 63
85 64 147 82 185 17 39 38 84 45 100 7 17 26 59 42 95 16 36 79 177 139 314 139
86 318 169 351 303 341 37 -3 65 -12 92 -30z"/>
87 <path d="M4395 4100 c-70 -36 -104 -86 -112 -165 1-6 -53 -261 0 c-143 0 -277
88 -3 -296 -7 -88 -15 -158 -102 -155 -195 2 -85 33 -137 106 -177 32 -17 70 -18
89 592 -20 307 -1 561 -3 563 -6 14 -13 -127 -249 -211 -352 -84 -105 -67 -105
90 -176 1 -103 99 -142 119 -217 111 -68 -8 -110 -34 -152 -96 -19 -27 -24 -48
91 -24 -95 0 -78 16 -108 106 -198 53 -53 67 -73 57 -79 -8 -5 -71 -38 -140 -74
92 -74 -38 -134 -76 -145 -93 -62 -84 -51 -202 24 -267 84 -73 153 -71 313 8 63
93 31 148 79 191 107 79 52 96 58 115 39 6 -6 20 -16 31 -23 11 -6 60 -35 108
94 -65 49 -30 105 -63 124 -74 19 -10 40 -23 45 -27 19 -15 109 -20 145 -9 91 30
95 141 104 136 201 -4 87 -36 126 -166 204 -58 35 -113 68 -123 74 -17 10 -14 16
96 30 68 73 87 113 140 113 151 0 5 3 11 8 13 10 4 62 85 62 97 0 5 4 11 8 13 17
97 7 134 254 166 351 3 9 16 17 28 18 108 3 187 79 193 184 4 67 -6 99 -43 141
98 -63 73 -78 75 -387 75 -82 0 -157 0 -165 0 -22 0 -151 0 -177 0 -13 -1 -23 5
99 -24 12 -3 86 -15 123 -56 166 -53 57 -160 76 -228 41z"/>
100 <path d="M1585 3398 c-4 -13 -22 -54 -39 -93 -18 -38 -48 -106 -67 -150 -19
101 -44 -41 -96 -51 -115 -9 -19 -15 -36 -13 -38 3 -4 367 -3 371 1 1 1 -22 56
102 -52 122 -76 170 -88 199 -86 208 1 4 -2 7 -6 7 -5 0 -15 18 -22 40 -15 45 -26
103 50 -35 18z"/>
104 </g>
105 </svg>
106

```

Translator/static/favicon/site.webmanifest

```

1 {
2   "name": "",
3   "short_name": "",
4   "icons": [
5     {
6       "src": "/android-chrome-192x192.png",
7       "sizes": "192x192",
8       "type": "image/png"
9     },
10    {
11      "src": "/android-chrome-512x512.png",
12      "sizes": "512x512",
13      "type": "image/png"
14    }
15  ],
16  "theme_color": "#ffffff",
17  "background_color": "#ffffff",
18  "display": "standalone"
19 }

```

Translator/translate/__init__.py

1

Translator/translate/__pycache__/views.cpython-39.pyc.140357439501584**Translator/translate/admin.py**

```

1 from django.contrib import admin
2
3 # Register your models here.
4

```

Translator/translate/apps.py

```

1 from django.apps import AppConfig
2
3
4 class TranslateConfig(AppConfig):
5     default_auto_field = 'django.db.models.BigAutoField'
6     name = 'translate'
7

```

Translator/translate/forms.py

```

1 from django import forms
2 from django.contrib.auth import authenticate
3 from django.contrib.auth.forms import UserCreationForm
4
5 from translate.models import User
6
7 import translator_backend
8
9
10 class LogInForm(forms.Form):
11     """Form to log in users. Adapted from the Clucker project from 5CCS2SEG."""
12     username = forms.CharField(label='Username', widget=forms.TextInput())
13     password = forms.CharField(label='Password', widget=forms.PasswordInput())
14
15     def get_user(self):
16         user = None
17         if self.is_valid():
18             username = self.cleaned_data.get('username')
19             password = self.cleaned_data.get('password')
20             user = authenticate(username=username, password=password)
21         return user
22
23
24 class SignUpForm(UserCreationForm):
25     class Meta:
26         model = User
27         fields = ['username', 'email']
28
29     def clean(self):
30         super().clean()
31         password = self.cleaned_data.get('password1')
32         password_confirmation = self.cleaned_data.get('password2')
33         if password != password_confirmation:
34             self.add_error('password1', 'Confirmation does not match password.')
35
36     def save(self, commit=True):
37         user = super().save(commit=False)
38         user.set_password(self.cleaned_data['password1'])
39         if commit:
40             user.save()

```

```

41         return user
42
43
44     class TranslatorForm(forms.Form):
45         language = forms.ChoiceField(label='Input language', choices=[('English', 'English'), ('Polish', 'Polish')])
46         text = forms.CharField(label='Text',
47                                widget=forms.Textarea(attrs={'rows': 10, 'cols': 50, 'placeholder': 'Enter text here'}))
48
49     def clean(self):
50         super().clean()
51         text = self.cleaned_data.get('text')
52         if not text:
53             self.add_error('text', 'Please enter some text.')
54
55     def translate(self):
56         text = self.cleaned_data.get('text')
57         if self.cleaned_data['language'] == 'English':
58             translated_texts = translator_backend.translateEnglishToPolish(text)
59         else:
60             translated_texts = translator_backend.translatePolishToEnglish(text)
61         whole_text = []
62         for translated_text in translated_texts:
63             whole_text.append(translated_text['translation_text'])
64         return '\n'.join(whole_text)
65

```

Translator/translate/migrations/0001_initial.py

```

# Generated by Django 4.1.5 on 2023-01-28 19:22
import django.contrib.auth.models
import django.contrib.auth.validators
from django.db import migrations, models
import django.utils.timezone


class Migration(migrations.Migration):
    initial = True

    dependencies = [
        ('auth', '0012_alter_user_first_name_max_length'),
    ]

    operations = [
        migrations.CreateModel(
            name='User',
            fields=[
                ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
                ('password', models.CharField(max_length=128, verbose_name='password')),
                ('last_login', models.DateTimeField(blank=True, null=True, verbose_name='last login')),
                ('is_superuser', models.BooleanField(default=False, help_text='Designates that this user has all permissions without explicitly assigning them.', verbose_name='superuser status')),
                ('username', models.CharField(error_messages={'unique': 'A user with that username already exists.'}, help_text='Required. 150 characters or fewer. Letters, digits and @./+/-/_ only.', max_length=150, unique=True, validators=[django.contrib.auth.validators.UnicodeUsernameValidator()], verbose_name='username')),
                ('first_name', models.CharField(blank=True, max_length=150, verbose_name='first name')),
                ('last_name', models.CharField(blank=True, max_length=150, verbose_name='last name')),
                ('is_staff', models.BooleanField(default=False, help_text='Designates whether the user can log into this admin site.', verbose_name='staff status')),
                ('is_active', models.BooleanField(default=True, help_text='Designates whether this user should be treated as active. Unselect this instead of deleting accounts.', verbose_name='active')),
                ('date_joined', models.DateTimeField(default=django.utils.timezone.now, verbose_name='date joined')),
                ('email', models.EmailField(max_length=254, unique=True)),
                ('groups', models.ManyToManyField(blank=True, help_text='The groups this user belongs to. A user will get all permissions granted to each of their groups.', related_name='user_set', related_query_name='user', verbose_name='groups')),
            ],
        ),
    ]

```

```

45         ('user_permissions', models.ManyToManyField(blank=True, help_text='Specific permissions for this
user.',
46                                                     related_name='user_set', related_query_name='user',
47                                                     to='auth.permission', verbose_name='user permissions')),
48     ],
49     options={
50         'verbose_name': 'user',
51         'verbose_name_plural': 'users',
52         'abstract': False,
53     },
54     managers=[
55         ('objects', django.contrib.auth.models.UserManager()),
56     ],
57 ),
58 ]
59

```

Translator/translate/migrations/0002_translation.py

```

1  # Generated by Django 4.1.5 on 2023-02-02 10:24
2
3  from django.conf import settings
4  from django.db import migrations, models
5  import django.db.models.deletion
6
7
8  class Migration(migrations.Migration):
9      dependencies = [
10         ('translate', '0001_initial'),
11     ]
12
13     operations = [
14         migrations.CreateModel(
15             name='Translation',
16             fields=[
17                 ('id', models.BigAutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
18                 ('text', models.CharField(max_length=255)),
19                 ('translated_text', models.CharField(max_length=255)),
20                 ('created_at', models.DateTimeField(auto_now_add=True)),
21                 ('updated_at', models.DateTimeField(auto_now=True)),
22                 ('user', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
to=settings.AUTH_USER_MODEL)),
23             ],
24         ),
25     ]
26

```

Translator/translate/migrations/0003_alter_translation_user.py

```

1  # Generated by Django 4.1.5 on 2023-02-02 11:38
2
3  from django.conf import settings
4  from django.db import migrations, models
5  import django.db.models.deletion
6
7
8  class Migration(migrations.Migration):
9      dependencies = [
10         ('translate', '0002_translation'),
11     ]
12
13     operations = [
14         migrations.AlterField(
15             model_name='translation',
16             name='user',
17             field=models.ForeignKey(null=True, on_delete=django.db.models.deletion.CASCADE,
to=settings.AUTH_USER_MODEL),
18         ),
19     ]
20
21

```

Translator/translate/migrations/0004_alter_translation_user.py

```

1 # Generated by Django 4.1.5 on 2023-02-02 11:45
2
3 from django.conf import settings
4 from django.db import migrations, models
5 import django.db.models.deletion
6
7
8 class Migration(migrations.Migration):
9     dependencies = [
10         ('translate', '0003_alter_translation_user'),
11     ]
12
13     operations = [
14         migrations.AlterField(
15             model_name='translation',
16             name='user',
17             field=models.ForeignKey(blank=True, null=True, on_delete=django.db.models.deletion.CASCADE,
18                                     to=settings.AUTH_USER_MODEL),
19         ),
20     ]
21

```

Translator/translate/migrations/0005_translation_input_language.py

```

1 # Generated by Django 4.1.5 on 2023-02-09 12:32
2
3 from django.db import migrations, models
4
5
6 class Migration(migrations.Migration):
7     dependencies = [
8         ('translate', '0004_alter_translation_user'),
9     ]
10
11     operations = [
12         migrations.AddField(
13             model_name='translation',
14             name='input_language',
15             field=models.CharField(choices=[('English', 'English'), ('Polish', 'Polish')], default='English',
16                                     max_length=7),
17             preserve_default=False,
18         ),
19     ]
20

```

Translator/translate/migrations/0006_remove_translation_created_at_and_more.py

```

1 # Generated by Django 4.1.5 on 2023-02-09 15:33
2
3 from django.db import migrations
4
5
6 class Migration(migrations.Migration):
7     dependencies = [
8         ('translate', '0005_translation_input_language'),
9     ]
10
11     operations = [
12         migrations.RemoveField(
13             model_name='translation',
14             name='created_at',
15         ),
16         migrations.RemoveField(
17             model_name='translation',
18             name='updated_at',
19         ),
20     ]
21

```

Translator/translate/migrations/0007_translation_created_at_translation_updated_at.py

```

1 # Generated by Django 4.1.5 on 2023-02-14 12:32

```

```

2
3 from django.db import migrations, models
4 import django.utils.timezone
5
6
7 class Migration(migrations.Migration):
8     dependencies = [
9         ('translate', '0006_remove_translation_created_at_and_more'),
10    ]
11
12    operations = [
13        migrations.AddField(
14            model_name='translation',
15            name='created_at',
16            field=models.DateTimeField(auto_now_add=True, default=django.utils.timezone.now),
17            preserve_default=False,
18        ),
19        migrations.AddField(
20            model_name='translation',
21            name='updated_at',
22            field=models.DateTimeField(auto_now=True),
23        ),
24    ]
25

```

Translator/translate/migrations/__init__.py

```

1

```

Translator/translate/models.py

```

1 from django.contrib.auth.models import AbstractUser
2 from django.db import models
3
4
5 class User(AbstractUser):
6     """A simple user model that extends the default Django user model."""
7     email = models.EmailField(max_length=254, unique=True)
8
9
10 class Translation(models.Model):
11     """A translation of a word or phrase."""
12
13     class LanguageChoices(models.TextChoices):
14         ENGLISH = 'English'
15         POLISH = 'Polish'
16
17     text = models.CharField(max_length=255)
18     translated_text = models.CharField(max_length=255)
19     user = models.ForeignKey(User, on_delete=models.CASCADE, null=True, blank=True)
20     input_language = models.CharField(max_length=7, choices=LanguageChoices.choices)
21     created_at = models.DateTimeField(auto_now_add=True)
22     updated_at = models.DateTimeField(auto_now=True)
23
24     def text_display(self):
25         return self.text[:27] + '...' if len(self.text) > 30 else self.text
26
27     def translated_text_display(self):
28         return self.translated_text[:27] + '...' if len(self.translated_text) > 30 else self.translated_text
29

```

Translator/translate/templates/base.html

```

1 {% load static %}
2 <!doctype html>
3 <html lang="en">
4 <head>
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1">
7     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
8         integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
        crossorigin="anonymous">

```

```

9     <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.6.1/font/bootstrap-icons.css">
10
11     <link href="{% static 'custom.css' %}" rel="stylesheet">
12     <link rel="apple-touch-icon" sizes="180x180" href="{% static 'favicon/apple-touch-icon.png' %}">
13     <link rel="icon" type="image/png" sizes="32x32" href="{% static 'favicon/favicon-32x32.png' %}">
14     <link rel="icon" type="image/png" sizes="16x16" href="{% static 'favicon/favicon-16x16.png' %}">
15     <link rel="manifest" href="{% static 'favicon/site.webmanifest' %}">
16     <link rel="mask-icon" href="{% static 'favicon/safari-pinned-tab.svg' %}" color="#320064">
17     <meta name="msapplication-TileColor" content="#fbf6ff">
18     <meta name="theme-color" content="ffffff">
19     <meta name="apple-mobile-web-app-title" content="Translator">
20     <meta name="application-name" content="Translator">
21
22     <script src="https://kit.fontawesome.com/29f2d97b0d.js" crossorigin="anonymous"></script>
23
24     <title>
25         Translator
26         {% block title %}
27         {% endblock %}
28     </title>
29
30     <meta property="og:url" content="{ request.build_absolute_uri }"/>
31     <meta property="og:type" content="website"/>
32     <meta property="og:title" content="Translate"/>
33     <meta property="og:description"
34         content="The implementation part of my final year project."/>
35
36     <style>
37         @import url('https://fonts.googleapis.com/css2?family=JetBrains+Mono:wght@300;400;500;700&display=swap');
38
39         * {
40             font-family: 'JetBrains Mono', sans-serif;
41         }
42     </style>
43 </head>
44 <body style="background-color: #282828; color: #FBF6FF">
45 <div id="background">
46     {% block body %}
47     {% endblock %}
48 </div>
49 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js"
50     integrity="sha384-QJHtvGhmr9X0IpI6VutG+2Q0K9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
51     crossorigin="anonymous"></script>
52 </body>
53

```

Translator/translate/templates/base_content.html

```

1  {% extends 'base.html' %}
2  {% block title %}
3  {% endblock %}
4  {% block body %}
5  {% include 'partials/navbar.html' %}
6  {% include 'partials/messages.html' %}
7  {% block content %}
8  {% endblock %}
9  {% endblock %}
10

```

Translator/translate/templates/home.html

```

1  {% extends 'base_content.html' %}
2  {% block title %}{% endblock %}
3  {% block content %}
4  {% load widget_tweaks %}
5  <div class="container" style="height: 100%">
6      <div class="row">
7          <div class="col-sm-12 col-md-6 offset-md-3">
8              <h1>Translator</h1>
9              {% if user.is_authenticated %}
10                 <p>Welcome, {{ user.username }}!</p>
11                 <a href="{ url 'logout' %}">Log out</a>
12             {% else %}
13                 <p>Welcome, guest!</p>
14                 <a href="{ url 'login' %}">Log in</a>

```

```

15         <a href="{% url 'signup' %}">Sign up</a>
16     {% endif %}
17     <p>Here you can translate text from one language to another.</p>
18 </div>
19 </div>
20 <br>
21 <!-- This is the translation form -->
22 <form action="{% url 'home' %}" method="post">
23     {% csrf_token %}
24     <div class="row" style="text-align: center">
25         <div class="col-sm-12 col-md-5">
26             {% include 'partials/bootstrap_form.html' with form=form %}
27         </div>
28
29         <div class="col-sm-12 col-md-2">
30             <input class="btn btn-secondary form-button" type="submit" value="Translate">
31         </div>
32
33         <div class="col-sm-12 col-md-5">
34             <label>Output in {{second_language}}
35             <textarea class="form-control" readonly="readonly" placeholder="Translated text"
36                 style="height: auto" rows="13" cols="50">{{ translations }}</textarea>
37             </label>
38         </div>
39     </div>
40 </form>
41
42
43 </div>
44 {% endblock %}

```

Translator/translate/templates/login.html

```

1  {% extends 'base_content.html' %}
2  {% block title %} | Log in{% endblock %}
3  {% block content %}
4  <div class="container">
5      <div class="row">
6          <div class="col-sm-12 col-md-6 offset-md-3 form-box">
7              <h1>Log in</h1>
8              <form action="{% url 'login' %}" method="post">
9                  {% csrf_token %}
10                 <input type="hidden" name="next" value="{{ next }}">
11                 {% include 'partials/bootstrap_form.html' with form=form %}
12                 <input class="btn btn-secondary form-button" type="submit" value="Log in">
13             </form>
14         </div>
15     </div>
16 </div>
17 {% endblock %}

```

Translator/translate/templates/partials/bootstrap_form.html

```

1  <!--This code was adapted from the 5CCS2SEG project Chess Club-->
2
3  {% load widget_tweaks %}
4  {% for field in form %}
5  <div class="mb-3">
6      {{ field.label_tag }}
7
8      {% if field.help_text %}
9      <p class="form-text text-muted">{{ field.help_text }}</p>
10     {% endif %}
11
12     {% if form.is_bound %}
13     {% if field.errors %}
14     {% render_field field class="form-control is-invalid" %}
15     {% else %}
16     {% render_field field class="form-control is-valid" %}
17     {% endif %}
18     {% else %}
19     {% render_field field class="form-control" %}
20     {% endif %}
21     <div class="valid-feedback"></div>
22     <div class="invalid-feedback">

```



```

23         {{ field.errors }}
24     </div>
25 </div>
26 {% endfor %}
27

```

Translator/translate/templates/partial/messages.html

```

1  <!--This code was adapted from the 5CCS2SEG project Chess Club-->
2
3  {% if messages %}
4  <div class="container">
5      <div class="row">
6          <div class="col-12">
7              {% for message in messages %}
8                  <div class="alert alert-{{ message.level_tag }}" role="alert">
9                      {{ message }}
10                 </div>
11             {% endfor %}
12         </div>
13     </div>
14 </div>
15 {% endif %}

```

Translator/translate/templates/partial/navbar.html

```

1  <!--This code was adapted from the 5CCS2SEG project Chess Club-->
2
3  <nav class="navbar navbar-expand-lg navbar-dark mb-3" style="background-color: #320064;">
4      <div class="container">
5
6          <a class="navbar-brand" href="{% url 'home' %}">
7              <span class="fas fa-solid fa-language"></span>&nbsp;&nbsp;&nbsp;Translate
8          </a>
9          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarSupportedContent"
10             aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
11              <span class="navbar-toggler-icon"></span>
12          </button>
13          <div class="collapse navbar-collapse" id="navbarSupportedContent">
14              <ul class="navbar-nav ms-auto mb-2 mb-lg-0">
15                  {% if user.is_authenticated %}
16                  <li class="nav-item dropdown">
17                      <a class="nav-link" href="#" id="user_account_dropdown" role="button" data-bs-toggle="dropdown"
18                         aria-expanded="false">
19                          <span class="bi-person-circle"></span>
20                      </a>
21                      <ul class="dropdown-menu dropdown-menu-end" aria-labelledby="user_account_dropdown">
22                          <li><a class="dropdown-item" href="{% url 'profile' user.id %}">View profile</a></li>
23                          <li><a class="dropdown-item" href="{% url 'password_change' %}">Change password</a></li>
24                          <li>
25                              <hr class="dropdown-divider">
26                          </li>
27                          <li><a class="dropdown-item" href="{% url 'logout' %}">Log out</a></li>
28                      </ul>
29                  </li>
30                  {% endif %}
31              </ul>
32          </div>
33      </div>
34  </nav>
35

```

Translator/translate/templates/password_change.html

```

1  {% extends 'base_content.html' %}
2  {% block title %} | Password change{% endblock %}
3  {% block content %}
4  <div class="container">
5      <div class="row">
6          <div class="col-sm-12 col-md-6 offset-md-3 form-box">
7              <h1>Change password</h1>

```

```

8         <form action="{% url 'password_change' %}" method="post">
9             {% csrf_token %}
10            <input type="hidden" name="next" value="{{ next }}">
11            {% include 'partials/bootstrap_form.html' with form=form %}
12            <input type="submit" value="Password change" class="btn btn-secondary form-button">
13        </form>
14    </div>
15</div>
16</div>
17{% endblock %}

```

Translator/translate/templates/profile.html

```

1  {% extends 'base_content.html' %}
2  {% block title %} | Profile{% endblock %}
3  {% block content %}
4  <div class="container">
5      <div class="row">
6          <div class="col-sm-12 col-md-6 offset-md-3">
7              <h1>User profile</h1>
8              <p>Username: {{ user.username }}</p>
9              <p>Email: {{ user.email }}</p>
10          </div>
11      </div>
12      <div class="row">
13          <div class="col-sm-12 col-md-8 offset-md-2 translation-box">
14              <!--Print all related translations-->
15              <div class="translation-item">
16                  <h4>History of translations:</h4>
17              </div>
18
19              {% for translation in translations %}
20              <div class="translation-item">
21                  <a class="text" href="{% url 'home' %}?translation={{translation.pk}}">{{ translation.text_display }}
22                  </div>
23              {% endfor %}
24          </div>
25      </div>
26  </div>
27  {% endblock %}

```

Translator/translate/templates/signup.html

```

1  {% extends 'base_content.html' %}
2  {% block title %} | Sign up{% endblock %}
3  {% block content %}
4  <div class="container">
5      <div class="row">
6          <div class="col-sm-12 col-md-6 offset-md-3 form-box">
7              <h1>Sign up</h1>
8              <form action="{% url 'signup' %}" method="post">
9                  {% csrf_token %}
10                 <input type="hidden" name="next" value="{{ next }}">
11                 {% include 'partials/bootstrap_form.html' with form=form %}
12                 <input type="submit" value="Sign up" class="btn btn-secondary form-button">
13             </form>
14         </div>
15     </div>
16</div>
17{% endblock %}

```

Translator/translate/tests/__init__.py

```

1 |

```

Translator/translate/tests/fixtures/default_translation.json

```

1 | [
2 | {

```

```

3     "model": "translate.translation",
4     "pk": 1,
5     "fields": {
6         "text": "Hello world!",
7         "input_language": "English",
8         "translated_text": "Witaj świat!",
9         "user": 1,
10        "created_at": "2023-02-14T12:00:00Z",
11        "updated_at": "2023-02-14T12:00:00Z"
12    }
13 }
14 ]

```

Translator/translate/tests/fixtures/default_user.json

```

1 [
2   {
3     "model": "translate.user",
4     "pk": 1,
5     "fields": {
6       "username": "MichaelScott",
7       "email": "michaelscott@dundermifflin.com",
8       "password": "pbkdf2_sha256$260000$VEDi9wsMYG6eNVeL8WSPqj$LHEiR2iUkusHCiEiQdWS+XQGC9/CjhhrjE0ESMMp+c0=",
9       "is_active": true
10    }
11  }
12 ]

```

Translator/translate/tests/fixtures/other_translations.json

```

1 [
2   {
3     "model": "translate.translation",
4     "pk": 2,
5     "fields": {
6       "text": "Witaj świecie!",
7       "input_language": "Polish",
8       "translated_text": "Hello world!",
9       "user": 1,
10      "created_at": "2023-02-14T12:00:00Z",
11      "updated_at": "2023-02-14T13:00:00Z"
12    }
13  },
14  {
15    "model": "translate.translation",
16    "pk": 3,
17    "fields": {
18      "text": "My name is Mikołaj and I like flowers.",
19      "input_language": "English",
20      "translated_text": "Nazywam się Mikołaj i lubię kwiaty.",
21      "user": 2,
22      "created_at": "2023-02-14T12:00:00Z",
23      "updated_at": "2023-02-14T15:00:00Z"
24    }
25  },
26  {
27    "model": "translate.translation",
28    "pk": 4,
29    "fields": {
30      "text": "Mój ulubiony kolor to czerwony. Jest to kolor miłości i zazdrości.",
31      "input_language": "Polish",
32      "translated_text": "My favorite color is red. It is the color of love and jealousy.",
33      "user": 2,
34      "created_at": "2023-02-14T12:00:00Z",
35      "updated_at": "2023-02-14T16:00:00Z"
36    }
37  },
38  {
39    "model": "translate.translation",
40    "pk": 5,
41    "fields": {
42      "text": "This is a very long text. It is long enough that it needs to be shortened to be displayed.",
43      "input_language": "English",
44      "translated_text": "To jest bardzo długi tekst. Jest wystarczająco długi, aby musiał zostać skrócony, aby
został wyświetlony.",

```

```

45         "user": 1,
46         "created_at": "2023-02-14T12:00:00Z",
47         "updated_at": "2023-02-14T18:00:00Z"
48     }
49 }
50 ]

```

Translator/translate/tests/fixtures/other_users.json

```

1  [
2  {
3      "model": "translate.user",
4      "pk": 2,
5      "fields": {
6          "username": "DwightSchrute",
7          "email": "dwightschrute@dundermifflin.com",
8          "password": "pbkdf2_sha256$260000$VEDi9wsMYG6eNveL8WSPqj$LHEiR2iUkushCieIQdWS+xQGC9/CjhhrjE0ESMMp+c0=",
9          "is_active": true
10     }
11 },
12 {
13     "model": "translate.user",
14     "pk": 3,
15     "fields": {
16         "username": "JimHalpert",
17         "email": "jimhalpert@dundermifflin.com",
18         "password": "pbkdf2_sha256$260000$VEDi9wsMYG6eNveL8WSPqj$LHEiR2iUkushCieIQdWS+xQGC9/CjhhrjE0ESMMp+c0=",
19         "is_active": true
20     }
21 },
22 {
23     "model": "translate.user",
24     "pk": 4,
25     "fields": {
26         "username": "PamBeesly",
27         "email": "pambeesly@dundermifflin.com",
28         "password": "pbkdf2_sha256$260000$VEDi9wsMYG6eNveL8WSPqj$LHEiR2iUkushCieIQdWS+xQGC9/CjhhrjE0ESMMp+c0=",
29         "is_active": true
30     }
31 }
32 ]

```

Translator/translate/tests/forms/__init__.py

```

1

```

Translator/translate/tests/forms/test_log_in_form.py

```

1  from django.test import TestCase
2
3  from django import forms
4  from translate.forms import LogInForm
5
6  """Tests of the log in form. The structure of the tests was inspired by the Clucker project from 5CCS2SEG."""
7
8
9  class TestLogInForm(TestCase):
10     fixtures = ['translate/tests/fixtures/default_user.json']
11
12     def setUp(self) -> None:
13         self.form_input = {'username': 'MichaelScott', 'password': 'Password123'}
14
15     def test_form_contains_required_fields(self):
16         form = LogInForm()
17         self.assertIn('username', form.fields)
18         self.assertIn('password', form.fields)
19         password_field = form.fields['password']
20         self.assertTrue(isinstance(password_field.widget, forms.PasswordInput))
21
22     def test_form_accepts_valid_input(self):
23         form = LogInForm(data=self.form_input)
24         self.assertTrue(form.is_valid())

```

```

25
26 def test_form_rejects_blank_username(self):
27     self.form_input['username'] = ''
28     form = LogInForm(data=self.form_input)
29     self.assertFalse(form.is_valid())
30     self.assertEqual(form.errors['username'], ['This field is required.'])
31
32 def test_form_rejects_blank_password(self):
33     self.form_input['password'] = ''
34     form = LogInForm(data=self.form_input)
35     self.assertFalse(form.is_valid())
36     self.assertEqual(form.errors['password'], ['This field is required.'])
37
38 def test_form_accepts_incorrect_username(self):
39     self.form_input['username'] = 'DwightSchrute'
40     form = LogInForm(data=self.form_input)
41     self.assertTrue(form.is_valid())
42
43 def test_form_accepts_incorrect_password(self):
44     self.form_input['password'] = 'WrongPassword123'
45     form = LogInForm(data=self.form_input)
46     self.assertTrue(form.is_valid())
47
48 def test_can_authenticate_valid_user(self):
49     form = LogInForm(data=self.form_input)
50     user = form.get_user()
51     self.assertEqual(user.username, 'MichaelScott')
52
53 def test_invalid_credentials_do_not_authenticate(self):
54     self.form_input['password'] = 'WrongPassword123'
55     form = LogInForm(data=self.form_input)
56     user = form.get_user()
57     self.assertEqual(user, None)
58
59 def test_blank_password_do_not_authenticate(self):
60     self.form_input['password'] = ''
61     form = LogInForm(data=self.form_input)
62     user = form.get_user()
63     self.assertEqual(user, None)
64
65 def test_blank_username_do_not_authenticate(self):
66     self.form_input['username'] = ''
67     form = LogInForm(data=self.form_input)
68     user = form.get_user()
69     self.assertEqual(user, None)
70

```

Translator/translate/tests/forms/test_sign_up_form.py

```

1 from django.test import TestCase
2
3 from django import forms
4 from translate.forms import SignUpForm
5
6 """Tests of the sign up form. The structure of the tests was inspired by the Clucker project from 5CCS2SEG."""
7
8
9 class TestSignUpForm(TestCase):
10     fixtures = ['translate/tests/fixtures/default_user.json']
11
12     def setUp(self):
13         self.form_input = {
14             'username': 'JimHalpert',
15             'email': 'jimhalpert@dundermifflin.com',
16             'password1': 'DwightIsTheBest123',
17             'password2': 'DwightIsTheBest123'
18         }
19
20     def test_form_contains_required_fields(self):
21         form = SignUpForm()
22         self.assertIn('username', form.fields)
23         self.assertIn('email', form.fields)
24         self.assertIn('password1', form.fields)
25         self.assertIn('password2', form.fields)
26         password1_field = form.fields['password1']
27         password2_field = form.fields['password2']

```

```

28     self.assertTrue(isinstance(password1_field.widget, forms.PasswordInput))
29     self.assertTrue(isinstance(password2_field.widget, forms.PasswordInput))
30
31     def test_form_accepts_valid_input(self):
32         form = SignUpForm(data=self.form_input)
33         self.assertTrue(form.is_valid())
34
35     def test_form_rejects_blank_username(self):
36         self.form_input['username'] = ''
37         form = SignUpForm(data=self.form_input)
38         self.assertFalse(form.is_valid())
39         self.assertEqual(form.errors['username'], ['This field is required.'])
40
41     def test_form_rejects_blank_email(self):
42         self.form_input['email'] = ''
43         form = SignUpForm(data=self.form_input)
44         self.assertFalse(form.is_valid())
45         self.assertEqual(form.errors['email'], ['This field is required.'])
46
47     def test_form_rejects_blank_password1(self):
48         self.form_input['password1'] = ''
49         self.form_input['password2'] = ''
50         form = SignUpForm(data=self.form_input)
51         self.assertFalse(form.is_valid())
52         self.assertEqual(form.errors['password1'], ['This field is required.'])
53
54     def test_form_rejects_blank_password2(self):
55         self.form_input['password2'] = ''
56         form = SignUpForm(data=self.form_input)
57         self.assertFalse(form.is_valid())
58         self.assertEqual(form.errors['password2'], ['This field is required.'])
59
60     def test_form_rejects_passwords_that_do_not_match(self):
61         self.form_input['password2'] = 'WrongPassword123'
62         form = SignUpForm(data=self.form_input)
63         self.assertFalse(form.is_valid())
64         self.assertEqual(form.errors['password2'], ['The two password fields didn't match.'])
65
66     def test_form_rejects_username_that_is_already_taken(self):
67         self.form_input['username'] = 'MichaelScott'
68         form = SignUpForm(data=self.form_input)
69         self.assertFalse(form.is_valid())
70         self.assertEqual(form.errors['username'], ['A user with that username already exists.'])
71
72     def test_form_rejects_email_that_is_already_taken(self):
73         self.form_input['email'] = 'michaelscott@dundermifflin.com'
74         form = SignUpForm(data=self.form_input)
75         self.assertFalse(form.is_valid())
76         self.assertEqual(form.errors['email'], ['User with this Email already exists.'])
77
78     def test_form_rejects_common_passwords(self):
79         self.form_input['password1'] = 'password'
80         self.form_input['password2'] = 'password'
81         form = SignUpForm(data=self.form_input)
82         self.assertFalse(form.is_valid())
83         self.assertEqual(form.errors['password2'], ['This password is too common.'])
84
85     def test_form_rejects_passwords_that_are_too_short(self):
86         self.form_input['password1'] = 'short'
87         self.form_input['password2'] = 'short'
88         form = SignUpForm(data=self.form_input)
89         self.assertFalse(form.is_valid())
90         self.assertEqual(form.errors['password2'],
91             ['This password is too short. It must contain at least 8 characters.'])
92
93     def test_form_rejects_passwords_similar_to_username(self):
94         self.form_input['password1'] = 'JimHalpert'
95         self.form_input['password2'] = 'JimHalpert'
96         form = SignUpForm(data=self.form_input)
97         self.assertFalse(form.is_valid())
98         self.assertEqual(form.errors['password2'], ['The password is too similar to the username.'])
99
100     def test_form_rejects_passwords_similar_to_email(self):
101         self.form_input['password1'] = 'dundermifflin.com'
102         self.form_input['password2'] = 'dundermifflin.com'
103         form = SignUpForm(data=self.form_input)
104         self.assertFalse(form.is_valid())
105         self.assertEqual(form.errors['password2'], ['The password is too similar to the email.'])
106

```

Translator/translate/tests/forms/test_translator_form.py

```
1 from django.test import TestCase
2
3 from translate.forms import TranslatorForm
4
5 """Tests of the translator form."""
6
7
8 class TestTranslatorForm(TestCase):
9     def setUp(self):
10         self.form_input = {
11             'text': 'Hello World',
12             'language': 'English'
13         }
14
15     def test_form_contains_required_fields(self):
16         form = TranslatorForm()
17         self.assertIn('text', form.fields)
18         self.assertIn('language', form.fields)
19
20     def test_form_accepts_valid_input(self):
21         form = TranslatorForm(data=self.form_input)
22         self.assertTrue(form.is_valid())
23
24     def test_form_rejects_blank_text(self):
25         self.form_input['text'] = ''
26         form = TranslatorForm(data=self.form_input)
27         self.assertFalse(form.is_valid())
28         self.assertEqual(form.errors['text'], ['This field is required.', 'Please enter some text.'])
29
30     def test_form_rejects_blank_language(self):
31         self.form_input['language'] = ''
32         form = TranslatorForm(data=self.form_input)
33         self.assertFalse(form.is_valid())
34         self.assertEqual(form.errors['language'], ['This field is required.'])
35
```

Translator/translate/tests/helpers.py

```
1 class LogInTester:
2     def _is_logged_in(self):
3         return '_auth_user_id' in self.client.session.keys()
4
```

Translator/translate/tests/models/__init__.py

```
1 |
```

Translator/translate/tests/models/test_translation_model.py

```
1 from django.core.exceptions import ValidationError
2 from django.test import TestCase
3 from translate.models import User, Translation
4
5 """Tests of the translation model."""
6
7
8 class TestTranslationModel(TestCase):
9     fixtures = ['translate/tests/fixtures/default_user.json']
10
11     def setUp(self):
12         self.user = User.objects.get(username='MichaelScott')
13         self.translation = Translation.objects.create(
14             user=self.user,
15             text='Hello',
16             translated_text='Cześć',
17             input_language='English'
18         )
19         self.translation.save()
```

```

20
21 def test_translation_is_valid(self):
22     self._assert_translation_is_valid()
23
24 def test_original_text_cannot_be_blank(self):
25     self.translation.text = ''
26     self._assert_translation_is_invalid()
27
28 def test_original_text_can_be_255_characters_long(self):
29     self.translation.text = 'x' * 255
30     self._assert_translation_is_valid()
31
32 def test_original_text_cannot_be_more_than_255_characters_long(self):
33     self.translation.text = 'x' * 256
34     self._assert_translation_is_invalid()
35
36 def test_translated_text_cannot_be_blank(self):
37     self.translation.translated_text = ''
38     self._assert_translation_is_invalid()
39
40 def test_translated_text_can_be_255_characters_long(self):
41     self.translation.translated_text = 'x' * 255
42     self._assert_translation_is_valid()
43
44 def test_translated_text_cannot_be_more_than_255_characters_long(self):
45     self.translation.translated_text = 'x' * 256
46     self._assert_translation_is_invalid()
47
48 def test_user_can_be_null(self):
49     self.translation.user = None
50     self._assert_translation_is_valid()
51
52 def test_input_language_cannot_be_blank(self):
53     self.translation.input_language = ''
54     self._assert_translation_is_invalid()
55
56 def test_input_language_can_only_be_english_or_polish(self):
57     self.translation.input_language = 'German'
58     self._assert_translation_is_invalid()
59
60 def _assert_translation_is_valid(self):
61     try:
62         self.translation.full_clean()
63     except ValidationError: # pragma: no cover
64         self.fail('Test translation should be valid')
65
66 def _assert_translation_is_invalid(self):
67     with self.assertRaises(ValidationError):
68         self.translation.full_clean()
69

```

Translator/translate/tests/models/test_user_model.py

```

1 from django.core.exceptions import ValidationError
2 from django.test import TestCase
3 from translate.models import User
4
5 """Tests of the user model. The structure of the tests was inspired by the clucker project from SCCS2SEG."""
6
7
8 class TestUserModel(TestCase):
9     fixtures = ['translate/tests/fixtures/default_user.json', 'translate/tests/fixtures/other_users.json']
10
11     def setUp(self):
12         self.user = User.objects.get(username='MichaelScott')
13
14     def test_user_is_valid(self):
15         self._assert_user_is_valid()
16
17     def test_username_cannot_be_blank(self):
18         self.user.username = ''
19         self._assert_user_is_invalid()
20
21     def test_username_must_be_unique(self):
22         dwight = User.objects.get(username='DwightSchrute')
23         self.user.username = dwight.username

```



```

24         self._assert_user_is_invalid()
25
26     def test_email_must_be_unique(self):
27         dwight = User.objects.get(username='DwightSchrute')
28         self.user.email = dwight.email
29         self._assert_user_is_invalid()
30
31     def test_username_can_be_150_characters_long(self):
32         self.user.username = 'x' * 150
33         self.test_user_is_valid()
34
35     def test_username_cannot_be_more_than_150_characters_long(self):
36         self.user.username = 'x' * 151
37         self._assert_user_is_invalid()
38
39     def test_email_must_contain_an_at(self):
40         self.user.email = 'michaelscottdundermifflin.com'
41         self._assert_user_is_invalid()
42
43     def test_email_must_contain_a_dot_in_the_domain(self):
44         self.user.email = 'michaelscott@dundermifflincom'
45         self._assert_user_is_invalid()
46
47     def test_email_can_be_254_characters_long(self):
48         self.user.email = 'x' * (254 - len("@dundermifflin.com")) + "@dundermifflin.com"
49         self._assert_user_is_valid()
50
51     def test_email_cannot_be_longer_than_254_characters_long(self):
52         self.user.email = 'x' * (255 - len("@dundermifflin.com")) + "@dundermifflin.com"
53         self._assert_user_is_invalid()
54
55     def _assert_user_is_valid(self):
56         try:
57             self.user.full_clean()
58         except ValidationError: # pragma: no cover
59             self.fail('Test user should be valid')
60
61     def _assert_user_is_invalid(self):
62         with self.assertRaises(ValidationError):
63             self.user.full_clean()
64

```

Translator/translate/tests/views/__init__.py

1 |

Translator/translate/tests/views/test_change_password_view.py

```

1  """Tests of the change password view."""
2  from django.test import TestCase
3  from django.urls import reverse
4
5  from translate.models import User
6
7
8  class HomeViewTestCase(TestCase):
9      """Tests of the change password view."""
10
11      fixtures = ['translate/tests/fixtures/default_user.json']
12
13      def setUp(self):
14          self.url = reverse('password_change')
15          self.user = User.objects.get(username='MichaelScott')
16          self.form = {"old_password": "Password123", "new_password1": "ThisIsMyPassword",
17                      "new_password2": "ThisIsMyPassword"}
18
19      def test_change_password_url(self):
20          self.assertEqual(self.url, '/password_change/')
21
22      def test_get_change_password(self):
23          self.client.login(username=self.user.username, password='Password123')
24          response = self.client.get(self.url)
25          self.assertEqual(response.status_code, 200)
26          self.assertTemplateUsed(response, 'password_change.html')
27

```

```

28 def test_post_change_password(self):
29     self.client.login(username=self.user.username, password='Password123')
30     response = self.client.post(self.url, self.form)
31     self.assertEqual(response.status_code, 302)
32     self.assertRedirects(response, '/')
33
34 def test_post_change_password_with_wrong_old_password(self):
35     self.client.login(username=self.user.username, password='Password123')
36     self.form['old_password'] = 'WrongPassword'
37     response = self.client.post(self.url, self.form)
38     self.assertEqual(response.status_code, 200)
39     self.assertTemplateUsed(response, 'password_change.html')
40
41 def test_post_change_password_with_blank_old_password(self):
42     self.client.login(username=self.user.username, password='Password123')
43     self.form['old_password'] = ''
44     response = self.client.post(self.url, self.form)
45     self.assertEqual(response.status_code, 200)
46     self.assertTemplateUsed(response, 'password_change.html')
47
48 def test_post_change_password_with_blank_new_password(self):
49     self.client.login(username=self.user.username, password='Password123')
50     self.form['new_password1'] = ''
51     response = self.client.post(self.url, self.form)
52     self.assertEqual(response.status_code, 200)
53     self.assertTemplateUsed(response, 'password_change.html')
54
55 def test_post_change_password_with_blank_confirm_password(self):
56     self.client.login(username=self.user.username, password='Password123')
57     self.form['new_password2'] = ''
58     response = self.client.post(self.url, self.form)
59     self.assertEqual(response.status_code, 200)
60     self.assertTemplateUsed(response, 'password_change.html')
61
62 def test_post_change_password_with_mismatched_passwords(self):
63     self.client.login(username=self.user.username, password='Password123')
64     self.form['new_password2'] = 'ThatIsNotMyPassword'
65     response = self.client.post(self.url, self.form)
66     self.assertEqual(response.status_code, 200)
67     self.assertTemplateUsed(response, 'password_change.html')
68

```

Translator/translate/tests/views/test_home_view.py

```

1  """Tests of the home view."""
2  from django.test import TestCase
3  from django.urls import reverse
4
5  from translate.models import User, Translation
6
7
8  class HomeViewTestCase(TestCase):
9      """Tests of the home view."""
10
11      fixtures = ['translate/tests/fixtures/default_user.json', 'translate/tests/fixtures/default_translation.json']
12
13      def setUp(self):
14          self.url = reverse('home')
15          self.user = User.objects.get(username='MichaelScott')
16          self.form = {'text': 'Hello world!', 'language': 'English'}
17
18      def test_home_url(self):
19          self.assertEqual(self.url, '/')
20
21      def test_get_home(self):
22          response = self.client.get(self.url)
23          self.assertEqual(response.status_code, 200)
24          self.assertTemplateUsed(response, 'home.html')
25
26      def test_get_home_with_translation(self):
27          response = self.client.get(self.url, {'translation': 1})
28          self.assertEqual(response.status_code, 200)
29          self.assertTemplateUsed(response, 'home.html')
30          self.assertInHTML('Hello world!', response.content.decode('utf-8'))
31
32      def test_post_home(self):

```

```

33     count_before = Translation.objects.count()
34     response = self.client.post(self.url, self.form)
35     count_after = Translation.objects.count()
36     self.assertEqual(response.status_code, 200)
37     self.assertTemplateUsed(response, 'home.html')
38     self.assertInHTML('Hello world!', response.content.decode('utf-8'))
39     self.assertInHTML('Witaj świecie! Witaj świecie!', response.content.decode('utf-8'))
40     self.assertEqual(count_after, count_before + 1)
41     translation = Translation.objects.last()
42     self.assertEqual(translation.text, 'Hello world!')
43     self.assertEqual(translation.translated_text, 'Witaj świecie! Witaj świecie!')
44     self.assertEqual(translation.user, None)
45
46     def test_post_home_with_an_existing_translation(self):
47         self.client.login(username=self.user.username, password='Password123')
48         count_before = Translation.objects.count()
49         response = self.client.post(self.url, self.form)
50         count_after = Translation.objects.count()
51         self.assertEqual(response.status_code, 200)
52         self.assertTemplateUsed(response, 'home.html')
53         self.assertInHTML('Hello world!', response.content.decode('utf-8'))
54         self.assertInHTML('Witaj świecie! Witaj świecie!', response.content.decode('utf-8'))
55         self.assertEqual(count_after, count_before)
56         translation = Translation.objects.last()
57         self.assertEqual(translation.text, 'Hello world!')
58         self.assertEqual(translation.translated_text, 'Witaj świecie! Witaj świecie!')
59         self.assertEqual(translation.user, self.user)
60
61     def test_post_home_with_polish(self):
62         count_before = Translation.objects.count()
63         response = self.client.post(self.url, {'text': 'Witaj świecie!', 'language': 'Polish'})
64         count_after = Translation.objects.count()
65         self.assertEqual(response.status_code, 200)
66         self.assertTemplateUsed(response, 'home.html')
67         self.assertInHTML('Witaj świecie!', response.content.decode('utf-8'))
68         self.assertInHTML('Hello world!', response.content.decode('utf-8'))
69         self.assertEqual(count_after, count_before + 1)
70         translation = Translation.objects.last()
71         self.assertEqual(translation.text, 'Witaj świecie!')
72         self.assertEqual(translation.translated_text, 'Hello world!')
73         self.assertEqual(translation.user, None)
74
75     def test_post_home_with_user(self):
76         self.client.login(username=self.user.username, password='Password123')
77         count_before = Translation.objects.count()
78         response = self.client.post(self.url, {'text': 'Hello there!', 'language': 'English'})
79         count_after = Translation.objects.count()
80         self.assertEqual(response.status_code, 200)
81         self.assertTemplateUsed(response, 'home.html')
82         self.assertInHTML('Hello there!', response.content.decode('utf-8'))
83         self.assertInHTML('Witajcie tutaj!', response.content.decode('utf-8'))
84         self.assertEqual(count_after, count_before + 1)
85         translation = Translation.objects.last()
86         self.assertEqual(translation.text, 'Hello there!')
87         self.assertEqual(translation.translated_text, 'Witajcie tutaj!')
88         self.assertEqual(translation.user, self.user)
89
90     def test_home_greets_guest(self):
91         response = self.client.get(self.url)
92         self.assertEqual(response.status_code, 200)
93         self.assertTemplateUsed(response, 'home.html')
94         self.assertInHTML('Welcome, guest!', response.content.decode('utf-8'))
95
96     def test_home_greets_user(self):
97         self.client.login(username=self.user.username, password='Password123')
98         response = self.client.get(self.url)
99         self.assertEqual(response.status_code, 200)
100        self.assertTemplateUsed(response, 'home.html')
101        self.assertInHTML('Welcome, MichaelScott!', response.content.decode('utf-8'))
102

```

Translator/translate/tests/views/test_log_in_view.py

```

1  from django.test import TestCase
2  from django.urls import reverse
3
4  from translate.models import User

```

```

5 from translate.tests.helpers import LogInTester
6
7 """Tests of the log in view. The structure of the tests was inspired by the Clucker project from 5CCS2SEG."""
8
9
10 class LogInViewTest(TestCase, LogInTester):
11     fixtures = ['translate/tests/fixtures/default_user.json']
12
13     def setUp(self):
14         self.url = reverse('login')
15         self.user = User.objects.get(username='MichaelScott')
16
17     def test_log_in_url(self):
18         self.assertEqual(self.url, '/login/')
19
20     def test_unsuccessful_log_in(self):
21         form_input = {'username': 'MichaelScott', 'password': 'WrongPassword123'}
22         response = self.client.post(self.url, form_input)
23         self.assertEqual(response.status_code, 200)
24         self.assertTemplateUsed(response, 'login.html')
25         self.assertFalse(self._is_logged_in())
26
27     def test_successful_log_in(self):
28         form_input = {'username': 'MichaelScott', 'password': 'Password123'}
29         response = self.client.post(self.url, form_input)
30         self.assertEqual(response.status_code, 302)
31         self.assertTrue(self._is_logged_in())
32         self.assertRedirects(response, reverse('home'))
33
34     def test_successful_log_in_with_redirect(self):
35         redirect_url = reverse('profile', kwargs={'pk': self.user.id})
36         form_input = {'username': 'MichaelScott', 'password': 'Password123', 'next': redirect_url}
37         response = self.client.post(self.url, form_input, follow=True)
38         self.assertTrue(self._is_logged_in())
39         self.assertRedirects(response, redirect_url, status_code=302, target_status_code=200)
40
41     def test_get_log_in(self):
42         response = self.client.get(self.url)
43         self.assertEqual(response.status_code, 200)
44         self.assertTemplateUsed(response, 'login.html')
45
46     def test_get_log_in_with_redirect(self):
47         redirect_url = reverse('profile', kwargs={'pk': self.user.id})
48         self.url = reverse('login') + '?next=' + redirect_url
49         response = self.client.get(self.url)
50         self.assertEqual(response.status_code, 200)
51         self.assertTemplateUsed(response, 'login.html')
52         self.assertEqual(response.context['next'], redirect_url)
53
54     def test_get_log_in_redirects_to_home_if_logged_in(self):
55         self.client.force_login(self.user)
56         response = self.client.get(self.url, follow=True)
57         self.assertRedirects(response, reverse('home'), status_code=302, target_status_code=200)
58         self.assertTemplateUsed(response, 'home.html')
59
60     def test_post_log_in_redirects_to_home_when_logged_in(self):
61         self.client.login(username=self.user.username, password='Password123')
62         form_input = {'username': 'wronguser', 'password': 'WrongPassword123'}
63         response = self.client.post(self.url, form_input, follow=True)
64         redirect_url = reverse('home')
65         self.assertRedirects(response, redirect_url, status_code=302, target_status_code=200)
66         self.assertTemplateUsed(response, 'home.html')
67

```

Translator/translate/tests/views/test_logout_view.py

```

1 """Tests of the log out view."""
2 from django.test import TestCase
3 from django.urls import reverse
4 from translate.models import User
5 from translate.tests.helpers import LogInTester
6
7
8 class LogOutViewTestCase(TestCase, LogInTester):
9     """Tests of the logout view."""
10
11     fixtures = ['translate/tests/fixtures/default_user.json']

```

```

12
13 def setUp(self):
14     self.url = reverse('logout')
15     self.user = User.objects.get(username='MichaelScott')
16
17 def test_log_out_url(self):
18     self.assertEqual(self.url, '/logout/')
19
20 def test_get_log_out(self):
21     self.client.login(username='MichaelScott', password='Password123')
22     self.assertTrue(self._is_logged_in())
23     response = self.client.get(self.url, follow=True)
24     response_url = reverse('home')
25     self.assertRedirects(response, response_url, status_code=302, target_status_code=200)
26     self.assertTemplateUsed(response, 'home.html')
27     self.assertFalse(self._is_logged_in())
28
29 def test_log_out_redirects_when_not_logged_in(self):
30     self.assertFalse(self._is_logged_in())
31     response = self.client.get(self.url, follow=True)
32     response_url = reverse('login') + '?next=' + self.url
33     self.assertRedirects(response, response_url, status_code=302, target_status_code=200)
34     self.assertTemplateUsed(response, 'login.html')
35     self.assertFalse(self._is_logged_in())
36

```

Translator/translate/tests/views/test_profile_view.py

```

1  """Tests of the show user view."""
2  from django.test import TestCase
3  from django.urls import reverse
4  from translate.models import User, Translation
5
6
7  class ShowUserViewTestCase(TestCase):
8      """Tests of the profile view."""
9
10     fixtures = [
11         'translate/tests/fixtures/default_user.json',
12         'translate/tests/fixtures/other_users.json',
13         'translate/tests/fixtures/default_translation.json',
14         'translate/tests/fixtures/other_translations.json',
15     ]
16
17     def setUp(self):
18         self.user = User.objects.get(username='MichaelScott')
19         self.target_user = User.objects.get(username='DwightSchrute')
20         self.url = reverse('profile', kwargs={'pk': self.user.id})
21
22     def test_show_user_url(self):
23         self.assertEqual(self.url, f'/profile/{self.user.id}/')
24
25     def test_get_show_user(self):
26         self.client.login(username=self.user.username, password='Password123')
27         response = self.client.get(self.url)
28         self.assertEqual(response.status_code, 200)
29         self.assertTemplateUsed(response, 'profile.html')
30
31     def test_get_show_user_redirects_when_not_logged_in(self):
32         redirect_url = reverse('login') + '?next=' + self.url
33         response = self.client.get(self.url)
34         self.assertRedirects(response, redirect_url, status_code=302, target_status_code=200)
35
36     def test_get_show_user_with_valid_id_fails_if_not_self(self):
37         self.client.login(username=self.user.username, password='Password123')
38         url = reverse('profile', kwargs={'pk': self.target_user.id})
39         response = self.client.get(url)
40         self.assertRedirects(response, reverse('home'), status_code=302, target_status_code=200)
41
42     def test_get_show_user_with_valid_id_of_self(self):
43         self.client.login(username=self.user.username, password='Password123')
44         response = self.client.get(self.url)
45         self.assertEqual(response.status_code, 200)
46         self.assertTemplateUsed(response, 'profile.html')
47         self.assertContains(response, "MichaelScott")
48         self.assertContains(response, "michaelscott@dundermifflin.com")

```

```

49         translations = Translation.objects.filter(user_id=self.user.id)
50         for translation in translations:
51             self.assertContains(response, translation.text_display() + ' -> ' +
translation.translated_text_display())
52
53     def test_get_show_user_with_invalid_id(self):
54         self.client.login(username=self.user.username, password='Password123')
55         url = reverse('profile', kwargs={'pk': self.user.id + 9999})
56         response = self.client.get(url)
57         self.assertRedirects(response, reverse('home'), status_code=302, target_status_code=200)
58

```

Translator/translate/tests/views/test_sign_up_view.py

```

1  from django.contrib.auth.hashers import check_password
2  from django.test import TestCase
3  from django.urls import reverse
4  from translate.forms import SignUpForm
5  from translate.models import User
6  from translate.tests.helpers import LogInTester
7
8  """Tests of the sign up view. The structure of the tests was inspired by the clucker project from 5CCS2SEG."""
9
10
11 class SignUpViewTestCase(TestCase, LogInTester):
12     fixtures = ['translate/tests/fixtures/default_user.json']
13
14     def setUp(self):
15         self.url = reverse('signup')
16         self.user = User.objects.get(username='MichaelScott')
17         self.form_input = {
18             'username': 'DwightSchrute',
19             'email': 'dwightschrute@dundermifflin.com',
20             'password1': 'JimHalpertIsTheBest123',
21             'password2': 'JimHalpertIsTheBest123'
22         }
23
24     def test_sign_up_url(self):
25         self.assertEqual(self.url, '/signup/')
26
27     def test_get_sign_up(self):
28         response = self.client.get(self.url)
29         self.assertEqual(response.status_code, 200)
30         self.assertTemplateUsed(response, 'signup.html')
31         form = response.context['form']
32         self.assertTrue(isinstance(form, SignUpForm))
33         self.assertFalse(form.is_bound)
34
35     def test_unsuccessful_sign_up(self):
36         self.form_input['username'] = 'BAD USERNAME'
37         before_count = User.objects.count()
38         response = self.client.post(self.url, self.form_input)
39         after_count = User.objects.count()
40         self.assertEqual(after_count, before_count)
41         self.assertEqual(response.status_code, 200)
42         self.assertTemplateUsed(response, 'signup.html')
43         form = response.context['form']
44         self.assertTrue(isinstance(form, SignUpForm))
45         self.assertTrue(form.is_bound)
46         self.assertFalse(self._is_logged_in())
47
48     def test_successful_sign_up(self):
49         before_count = User.objects.count()
50         response = self.client.post(self.url, self.form_input, follow=True)
51         response_url = reverse('home')
52         self.assertRedirects(response, response_url, status_code=302, target_status_code=200)
53         after_count = User.objects.count()
54         self.assertEqual(after_count, before_count + 1)
55         self.assertTemplateUsed(response, 'home.html')
56
57         user = User.objects.get(username='DwightSchrute')
58         self.assertEqual(user.email, 'dwightschrute@dundermifflin.com')
59         is_password_correct = check_password(self.form_input['password1'], user.password)
60         self.assertTrue(is_password_correct)
61         self.assertTrue(self._is_logged_in())
62
63     def test_get_sign_up_redirects_to_home_when_logged_in(self):

```

```

64         self.client.login(username=self.user.username, password='Password123')
65         response = self.client.get(self.url, follow=True)
66         redirect_url = reverse('home')
67         self.assertTemplateUsed(response, 'home.html')
68         self.assertRedirects(response, redirect_url, status_code=302, target_status_code=200)
69
70     def test_post_sign_up_redirects_to_home_when_logged_in(self):
71         before_count = User.objects.count()
72         self.client.login(username=self.user.username, password='Password123')
73         response = self.client.post(self.url, self.form_input, follow=True)
74         after_count = User.objects.count()
75         self.assertEqual(after_count, before_count)
76         redirect_url = reverse('home')
77         self.assertTemplateUsed(response, 'home.html')
78         self.assertRedirects(response, redirect_url, status_code=302, target_status_code=200)
79

```

Translator/translate/views.py

```

1  from django.contrib.auth import login
2  from django.contrib.auth.forms import PasswordChangeForm, PasswordResetForm
3  from django.core.exceptions import ImproperlyConfigured
4  from django.http import Http404
5  from django.shortcuts import render, redirect
6  from django.contrib.auth.views import LogoutView
7  from django.contrib.auth.mixins import LoginRequiredMixin
8  from django.views import View
9  from django.views.generic import DetailView, CreateView
10 from django.urls import reverse
11 from django.contrib import messages
12
13 from translate.forms import LogInForm, SignUpForm, TranslatorForm
14 from translate.models import User, Translation
15 from translator import settings
16
17
18 class LoginProhibitedMixin:
19     """The LoginProhibitedMixin is used to prevent logged in users from accessing the login and signup pages.
20     It is adapted from the Clucker project from 5CCS2SEG."""
21     redirect_when_logged_in_url = None
22
23     def dispatch(self, *args, **kwargs):
24         if self.request.user.is_authenticated:
25             return self.handle_already_logged_in(*args, **kwargs)
26         return super().dispatch(*args, **kwargs)
27
28     def handle_already_logged_in(self, *args, **kwargs):
29         url = self.get_redirect_when_logged_in_url()
30         return redirect(url)
31
32     def get_redirect_when_logged_in_url(self):
33         if self.redirect_when_logged_in_url is None:
34             raise ImproperlyConfigured('LoginProhibitedMixin requires either a value for ' # pragma: no cover
35                                     'redirect_when_logged_in_url or an implementation '
36                                     'for get_redirect_when_logged_in_url().')
37         else:
38             return self.redirect_when_logged_in_url
39
40
41 class HomeView(View):
42     """View that shows the home page."""
43     template_name = 'home.html'
44     translated_text = ''
45     text = ''
46     form = TranslatorForm()
47
48     def get(self, request):
49         translation_pk = request.GET.get('translation', None)
50         if translation_pk:
51             translation = Translation.objects.get(pk=translation_pk)
52             self.form = TranslatorForm(initial={'text': translation.text, 'language': translation.input_language})
53
54         return self.render()
55
56     def post(self, request):
57         self.form = TranslatorForm(request.POST)

```

```

58     if self.form.is_valid():
59         self.translated_text = self.form.translate()
60
61         # try to find an existing translation
62         try:
63             translation = Translation.objects.get(
64                 text=self.form.cleaned_data['text'],
65                 input_language=self.form.cleaned_data['language'],
66                 user_id=request.user.pk
67             )
68             translation.translated_text = self.translated_text
69             translation.save()
70         except Translation.DoesNotExist:
71             # create a new translation
72             translation = Translation.objects.create(
73                 text=self.form.cleaned_data['text'],
74                 translated_text=self.translated_text,
75                 user_id=request.user.pk,
76                 input_language=self.form.cleaned_data['language']
77             )
78
79             translation.save()
80             self.text = self.form.cleaned_data['text']
81
82     return self.render()
83
84     def render(self):
85         return render(self.request, self.template_name,
86                       {'form': self.form, 'text': self.text, 'translations': self.translated_text,
87                       'second_language': self.get_second_language()})
88
89     def get_second_language(self):
90         try:
91             if self.form.cleaned_data['language'] == 'English':
92                 return 'Polish'
93             else:
94                 return 'English'
95         except AttributeError:
96             return 'Polish'
97
98
99 class LoginView(LoginProhibitedMixin, View):
100     """View that logs in user. The structure was inspired by the Clucker project from 5CCS2SEG."""
101     template_name = 'login.html'
102     next_page = '/'
103     form_class = LoginForm
104     http_method_names = ['get', 'post']
105     redirect_when_logged_in_url = settings.LOGIN_REDIRECT_URL
106
107     def get(self, request):
108         self.next_page = request.GET.get('next') or '/'
109         return self.render()
110
111     def post(self, request):
112         form = self.form_class(request.POST)
113         self.next_page = request.POST.get('next') or settings.LOGIN_REDIRECT_URL
114         user = form.get_user()
115         if user is not None:
116             login(request, user)
117             return redirect(self.next_page)
118         messages.add_message(request, messages.ERROR, 'The provided credentials were invalid!')
119         return self.render()
120
121     def render(self):
122         return render(self.request, self.template_name, {'form': self.form_class, 'next': self.next_page})
123
124
125 class LogoutView(LoginRequiredMixin, LogoutView):
126     next_page = '/'
127     success_url = '/'
128     template_name = 'logout.html'
129     login_url = '/login/'
130
131
132 class SignUpView(LoginProhibitedMixin, CreateView):
133     form_class = SignUpForm
134     template_name = 'signup.html'
135     redirect_when_logged_in_url = settings.LOGIN_REDIRECT_URL
136     success_url = settings.LOGIN_REDIRECT_URL

```



```

137
138     def form_valid(self, form):
139         form.save()
140         login(self.request, form.instance)
141         return redirect(self.get_success_url())
142
143     def get_success_url(self):
144         return reverse(self.success_url)
145
146
147 class ProfileView(LoginRequiredMixin, DetailView):
148     model = User
149     template_name = 'profile.html'
150     login_url = '/login/'
151
152     def get_context_data(self, **kwargs):
153         context = super().get_context_data(**kwargs)
154         user = self.get_object()
155         # can only view your own profile
156         if user != self.request.user:
157             raise Http404
158         translations = Translation.objects.filter(user=user)
159         # sort translations by date
160         translations = translations.order_by('-updated_at')
161         context['translations'] = translations
162         return context
163
164     def get(self, request, *args, **kwargs):
165         try:
166             return super(ProfileView, self).get(request, *args, **kwargs)
167         except Http404:
168             return redirect('home')
169
170
171 class PasswordChangeView(LoginRequiredMixin, View):
172     template_name = 'password_change.html'
173     success_url = '/'
174     form_class = PasswordChangeForm
175
176     def get(self, request):
177         form = self.form_class(request.user)
178         return render(request, self.template_name, {'form': form})
179
180     def post(self, request):
181         form = self.form_class(request.user, request.POST)
182         if form.is_valid():
183             user = form.save()
184             login(request, user)
185             return redirect(self.success_url)
186         messages.add_message(request, messages.ERROR, 'The provided credentials were invalid!')
187         return render(request, self.template_name, {'form': form})
188

```

Translator/translator/__init__.py

1 |

Translator/translator/asgi.py

```

1  """
2  ASGI config for translator project.
3
4  It exposes the ASGI callable as a module-level variable named ``application``.
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/4.1/howto/deployment/asgi/
8  """
9
10 import os
11
12 from django.core.asgi import get_asgi_application
13
14 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'translator.settings')
15
16 application = get_asgi_application()

```

Translator/translator/settings.py

```

1  """
2  Django settings for translator project.
3
4  Generated by 'django-admin startproject' using Django 4.1.5.
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/4.1/topics/settings/
8
9  For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/4.1/ref/settings/
11 """
12 from os import path
13 from pathlib import Path
14 from django.contrib.messages import constants as message_constants
15
16 # Build paths inside the project like this: BASE_DIR / 'subdir'.
17 BASE_DIR = Path(__file__).resolve().parent.parent
18
19
20 # Quick-start development settings - unsuitable for production
21 # See https://docs.djangoproject.com/en/4.1/howto/deployment/checklist/
22
23 # SECURITY WARNING: keep the secret key used in production secret!
24 SECRET_KEY = 'django-insecure-2_3%pqhek9kyo-!wx@okx4j%ueq#%fq91$1$q=j&9a=a$49z@'
25
26 # SECURITY WARNING: don't run with debug turned on in production!
27 DEBUG = True
28
29 ALLOWED_HOSTS = []
30
31
32 # Application definition
33
34 INSTALLED_APPS = [
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41     'translate',
42     'widget_tweaks',
43     'django_extensions',
44 ]
45
46 MIDDLEWARE = [
47     'django.middleware.security.SecurityMiddleware',
48     'django.contrib.sessions.middleware.SessionMiddleware',
49     'django.middleware.common.CommonMiddleware',
50     'django.middleware.csrf.CsrfViewMiddleware',
51     'django.contrib.auth.middleware.AuthenticationMiddleware',
52     'django.contrib.messages.middleware.MessageMiddleware',
53     'django.middleware.clickjacking.XFrameOptionsMiddleware',
54 ]
55
56 ROOT_URLCONF = 'translator.urls'
57
58 TEMPLATES = [
59     {
60         'BACKEND': 'django.template.backends.django.DjangoTemplates',
61         'DIRS': [],
62         'APP_DIRS': True,
63         'OPTIONS': {
64             'context_processors': [
65                 'django.template.context_processors.debug',
66                 'django.template.context_processors.request',
67                 'django.contrib.auth.context_processors.auth',
68                 'django.contrib.messages.context_processors.messages',
69             ],
70         },
71     },
72 ]
73

```

```

74 WSGI_APPLICATION = 'translator.wsgi.application'
75
76
77 # Database
78 # https://docs.djangoproject.com/en/4.1/ref/settings/#databases
79
80 DATABASES = {
81     'default': {
82         'ENGINE': 'django.db.backends.sqlite3',
83         'NAME': BASE_DIR / 'db.sqlite3',
84     }
85 }
86
87
88 # Password validation
89 # https://docs.djangoproject.com/en/4.1/ref/settings/#auth-password-validators
90
91 AUTH_PASSWORD_VALIDATORS = [
92     {
93         'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
94     },
95     {
96         'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
97     },
98     {
99         'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
100     },
101     {
102         'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
103     },
104 ]
105
106
107 # Internationalization
108 # https://docs.djangoproject.com/en/4.1/topics/i18n/
109
110 LANGUAGE_CODE = 'en-us'
111
112 TIME_ZONE = 'UTC'
113
114 USE_I18N = True
115
116 USE_TZ = True
117
118
119 # Static files (CSS, JavaScript, Images)
120 # https://docs.djangoproject.com/en/4.1/howto/static-files/
121
122 STATIC_URL = '/static/'
123 STATICFILES_DIRS = [
124     path.join(BASE_DIR, 'static')
125 ]
126
127 # Default primary key field type
128 # https://docs.djangoproject.com/en/4.1/ref/settings/#default-auto-field
129
130 DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
131
132 # User model for the purpose of authentication
133 AUTH_USER_MODEL = 'translate.User'
134
135 APPEND_SLASH = False
136
137 LOGIN_REDIRECT_URL = 'home'
138
139 # Message level tags should use Bootstrap terms
140 MESSAGE_TAGS = {
141     message_constants.DEBUG: 'dark',
142     message_constants.ERROR: 'danger',
143 }
144

```

Translator/translator/urls.py

```

1| """translator URL Configuration
2|
3| The `urlpatterns` list routes URLs to views. For more information please see:

```

```

4     https://docs.djangoproject.com/en/4.1/topics/http/urls/
5 Examples:
6 Function views
7     1. Add an import: from my_app import views
8     2. Add a URL to urlpatterns: path('', views.home, name='home')
9 Class-based views
10    1. Add an import: from other_app.views import Home
11    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13    1. Import the include() function: from django.urls import include, path
14    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18
19 from translate import views
20
21 urlpatterns = [
22     path('admin/', admin.site.urls),
23     path('', views.HomeView.as_view(), name='home'),
24     path('accounts/', include('django.contrib.auth.urls')),
25     path('signup/', views.SignUpView.as_view(), name='signup'),
26     path('login/', views.LogInView.as_view(), name='login'),
27     path('profile/<int:pk>', views.ProfileView.as_view(), name='profile'),
28     path('password_change/', views.PasswordChangeView.as_view(), name='password_change'),
29     path('logout/', views.LogOutView.as_view(), name='logout'),
30 ]
31

```

Translator/translator/wsgi.py

```

1 """
2 WSGI config for translator project.
3
4 It exposes the WSGI callable as a module-level variable named ``application``.
5
6 For more information on this file, see
7 https://docs.djangoproject.com/en/4.1/howto/deployment/wsgi/
8 """
9
10 import os
11
12 from django.core.wsgi import get_wsgi_application
13
14 os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'translator.settings')
15
16 application = get_wsgi_application()
17

```

Translator/translator_backend.py

```

1 from transformers import pipeline
2
3 print("Loading models...")
4 translator_en_pl = pipeline('translation', tokenizer="alirezamsh/small100",
5                             model="MikolajDeja/alirezamsh-small100-en-pl-para_crawl-finetune",
6                             src_lang="en", tgt_lang="pl")
7
8 translator_pl_en = pipeline("translation", model="MikolajDeja/Helsinki-NLP-opus-mt-pl-en-para_crawl-finetune",
9                             tokenizer="Helsinki-NLP/opus-mt-pl-en")
10
11 print("Models loaded.")
12
13
14 def translateEnglishToPolish(english_sentence):
15     return translator_en_pl(english_sentence.split("\n"))
16
17
18 def translatePolishToEnglish(polish_sentence):
19     return translator_pl_en(polish_sentence.split("\n"))
20

```