IN5550 – Spring 2024 — Obligatory 2
Cross-lingual Named Entity Recognition

Fernando Vallecillos Ruiz and **Sushant Gautam**

2024/03/15

0 Dataset and pre-processing

0.1 Pre-processing

Although we had employed heavy prepossessing on the previous assignment, we decided not to apply those to this assignment. The choice was directed by LLM's capabilities for contextual understanding and its subword tokenization, which made us believe that it is capable of handling raw text without preprocessing.

0.2 Experimentation

We created a function that facilitates experimentation with various train/test language and hyperparameters such as learning rate, batch size, and fine-tuning by allowing these parameters to be easily adjusted as arguments. This flexibility enabled us to conduct experiments efficiently by iterating over different parameter configurations without modifying the core training and evaluation logic. Inside the function, a tokenizer and a token classification model are initialized based on the provided **model_path**. Training and validation datasets are combined and split based on the provided language list (train_langs and val_langs). We implement our parsing script and are also aware that *datasets.load_dataset("xtreme", data_dir=...")* can do the job. These datasets are then tokenized and converted into PyTorch DataLoader objects. The model is trained using the train function that implements the training loop, including backpropagation and parameter updates, with the training DataLoader. After training, the function iterates over the validation languages (val_langs) to evaluate the model's performance (F1) on each language. Test datasets for each language are loaded, tokenized, and converted into DataLoader objects. The function then evaluates the model's classification performance on the test data, calculating loss and evaluation scores (F1; per class and aggregated; strict IOB2). Finally, training and validation loss curve is plotted along with validation F1 curve. We used *seqeval*'s *classification_report* and *f1_score* functions to get the metrics.

1 Part 1: Different Language Models

1.1 Data Splitting

The training and validation datasets were obtained using the '*train*' set and the test set was obtained from '*dev*' set for each language. The given '*train*' set was splitted into two parts: a training set (80%) and a validation set (20%) using the *train_test_split* function from a *scikit-learn*. Stratification strategy was used to ensure that both subsets maintain a similar distribution of sentence lengths, which can be crucial for training models on textual data where the length of sentences might influence model performance. The sentences were categorized into predefined bins (ranging from '0-5' words to '40-100' words) based on their length. This categorization process ensures that sentences are grouped based on their length, which is an indirect way of maintaining diversity and representativeness in both the training and test sets. However, it must be said that sometimes the bins chosen do not hold enough samples (at least 2), which results in errors in the code. Therefore, we chose to drop said bins and assume that loss is minimal.

Set	Length	Tokens	Unique	O	I-ORG	I-PER	I-LOC	B-LOC	B-ORG	B-PER
Train	16000	128429	28681	65017	18842	11628	10530	7536	7623	7253
Val	4000	31965	10319	16345	4384	3070	2647	1809	1799	1911
Test	10000	80536	20222	40875	11638	7520	6357	4834	4677	4635

Table 1: Dataset split overview with named entity distributions.

1.2 Hyperparameter Optimization

In our experimentation phase, we explored various hyperparameters to optimise our models' performance. The primary hyperparameters under consideration included the learning rate, batch size, number of epochs, and the implementation of early stopping to prevent overfitting.

- **Learning Rate:** A fixed rate of 1×10^{-5} was chosen for nearly all experiments, providing stable convergence of loss curves. Ablation study with explicit LR variations has been done on 1.7.1.
- **Batch Size:** Set to 128 for all the experiments/models except DeBerta (64 for all experiments with DeBerta) to manage GPU memory constraints effectively, this size was found to be optimal.
- **Number of Epochs:** Limited to 20 (unless explicitly specified), with early stopping based on validation loss to prevent overtraining.
- **Early Stopping:** Implemented with a patience of 3 epochs, this technique halted training when no improvement in validation loss was observed, enhancing generalization.
- **LR Scheduler**: ReduceLROnPlateau scheduler was applied to adjust the learning rate based on validation loss, with a patience of 3 epochs and a reduction factor of 0.1.
- **Optimizer**: `AdamW` was utilized with default settings, and weight decay adjustments were explored but ultimately deemed unnecessary due to a lack of overfitting issues.
- **Gradient Clipping:** Experimented with few value but found no benefit on training. As, its impact on model performance was minimal, leading to its exclusion in subsequent experiments.

1.3 Model Selection

To select a third encoder-only transformer model, we have tried a different selection on models. The models are chosen following the state-of-the-art in NER. Since the quantity and variation makes it unfeasible to try all of them, we have chosen two main models, named Electra and DeBerta. Given the success of DeBerta on the initial findings, we continue exploring related models. Table 2 relates the name of the model in this following with its uniquely HuggingFace ID.

Model	HuggingFace Model
Electra	google/electra-base-discriminator
DeBerta	microsoft/deberta-base
DeBerta v3	microsoft/deberta-v3-base
M-DeBerta v3	microsoft/mdeberta-v3-base
NER M-DeBerta v3	Grpp/rured2-ner-mdeberta-v3-base

Table 2: Correspondence between models and HuggingFace IDs.

As mentioned previously, given the initial success of the DeBerta model, we followed by using its latest version (DeBerta v3). We then followed by using the multilingual version which will help obtain a more general model. Finally, we chose a version already fine-tuned on NER tasks.

We have the train and validation set provided only for the English language. This decision was made to reduce the training and testing time. The training and validation loss through the epoch can be seen in the following graphs.

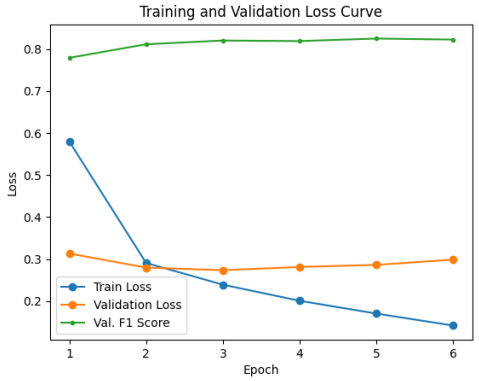

Figure 1: Loss and F1 curve for M-BERT-Cased trained with English corpus only for evaluating performance.

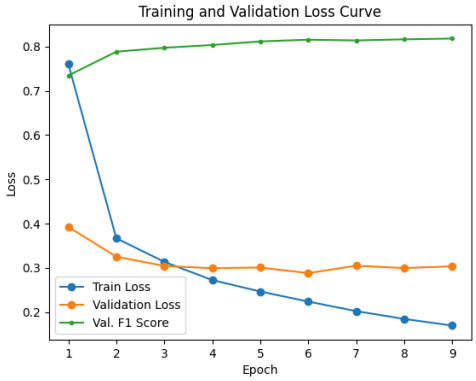

Figure 2: Loss and F1 curve for XLM-Roberta trained with English corpus only for evaluating performance.

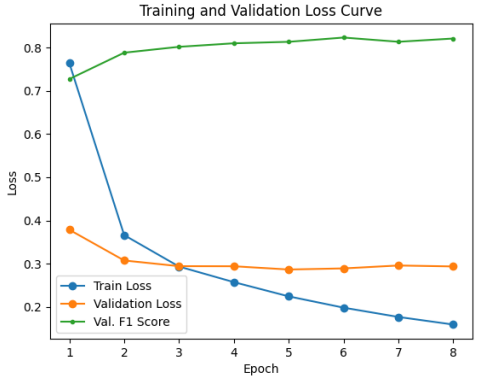

Figure 3: Loss and F1 curve for Electra trained with English corpus only for evaluating performance.

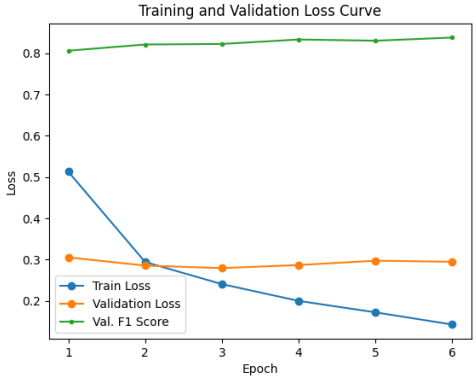

Figure 4: Loss and F1 curve for DeBerta trained with English corpus only for evaluating performance.

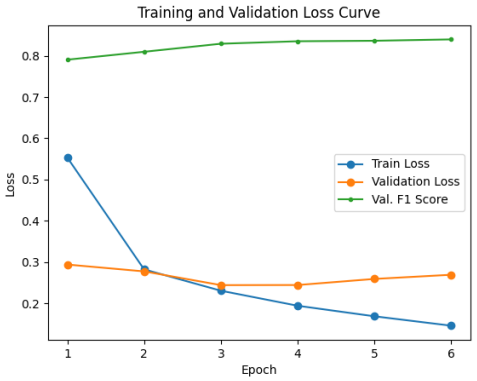

Figure 5: Loss and F1 curve for DeBerta v3 trained with English corpus only for evaluating performance.

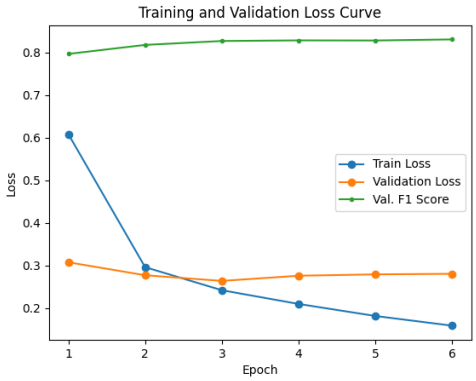

Figure 6: Loss and F1 curve for M-DeBerta v3 trained with English corpus only for evaluating performance.

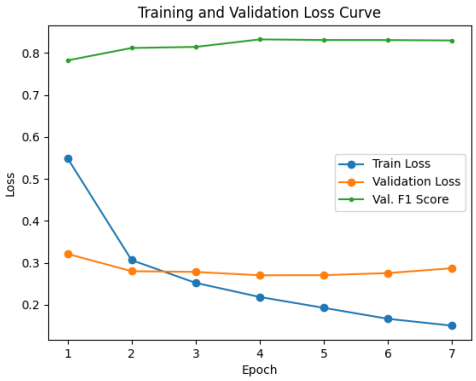

Figure 7: Loss and F1 curve for NER M-DeBerta v3 trained with English corpus only for evaluating performance.

We can appreciate that the training and validation loss tend to converge during the second and third epoch. XLM-Roberta and Electra are the models which converge the latest. Both models also start with the highest amount of training and validation loss among the models chosen. The multilingual version of DeBerta v3 has an initial increased loss compared to its counterpart. This follows the trend of multilingual version being more general, or able to handle more languages, but their abilities for specific languages deteriorates the more languages are added to their training.

We also summarize the number of epochs required, the Macro F1-Score and the time taken for training each of the models in table 3.

Model	Epochs	Macro F1-Score	Time Elapsed
M-Bert Cased	6	0.837	1016 s
XLM Roberta	9	0.831	1513 s
Electra	8	0.828	1318 s
DeBerta	6	0.848	1537 s
DeBerta v3	6	0.845	1468 s
M-DeBerta v3	6	0.849	1506 s
NER M-DeBerta v3	7	0.845	1740 s

Table 3: Results obtained from models fine-tuned and validated on English dataset.

1.4 Training/Inference Speed

Based on Table 4, other models exhibit faster training times compared to **mdeberta**. Similarly, their inference times are also shorter. This suggests that the former models are more efficient in both the training and inference stages. The reason behind this efficiency could be attributed to parameter size.

Table 4: Training (single epoch) and Inference Time Comparison; batch size 64 for both training (including validation metric calculation) and inference. Used **16,000** English samples both for training and the same for inference. Tested on *NVIDIA GeForce RTX 3090* on node: *gpu-4.fox* on Fox.

Model	Training Time	Inference Time
`mdeberta-v3-base`	111.22s	37.16s
`rured2-ner-mdeberta-v3-base`	110.58	36.98s
`xlm-roberta-base`	82.89s	28.46s
`bert-base-multilingual-cased`	80.36s	29.53s
`deberta-base`	78.64s	29.83s
`electra-base-discriminator`	78.50s	29.56s

1.5 Performance across languages

It was seen that the effectiveness of cross-lingual transfer learning for NER varies significantly across languages largely influenced by linguistic similarities. The model's high performance in languages like Italian and its struggles with Afrikaans and Swahili highlight the importance of linguistic diversity in training data and the need for language-specific adaptation strategies to improve NER performance across a broader range of languages.

Performance Summary

1. English (en): Highest overall performance with a Token-level F1-score of 0.828. Shows strong performance across all entity types, especially in recognizing PER (People) entities.
2. Italian (it): The second-highest performing language with a Token-level F1-score of 0.769. Notably high performance in PER entity recognition, slightly outperforming English in precision and recall for this category.
3. German (de): Moderate performance with a Token-level F1-score of 0.701. Shows a balanced performance across LOC and PER but struggles with ORG entities.
4. Swahili (sw): Lower middle performance with a Token-level F1-score of 0.691. Despite lower aggregate scores, it performs remarkably well in PER entity recognition.
5. Afrikaans (af): Lowest performance with a Token-level F1-score of 0.665. Struggles significantly with ORG entity recognition, though it does reasonably well with LOC and PER entities

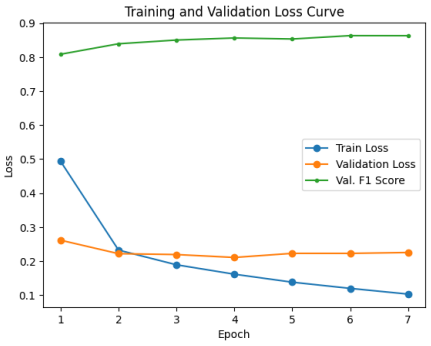

Figure 8: Loss and F1 curve for model trained with English corpus only for evaluating performance.

Language	LOC	ORG	PER	**Macro F1-Score**
English (en)	**0.875**	**0.739**	**0.888**	**0.828**
Italian (it)	0.791	0.641	0.895	0.769
Afrikaans (af)	0.762	0.534	0.778	0.665
Swahili (sw)	0.752	0.457	0.840	0.691
German (de)	0.792	0.573	0.744	0.701

Figure 9: NER Model Performance Across Different Languages for *bert-base-multilingual-cased* trained for 4 epoch for Figure 8

Analysis and Interpretation The model performs best in English, unsurprisingly, as it's the language it was trained on. Italian with considerable lexical similarity to English, shows relatively high performance. German has more complex syntax and compounding of nouns, which may explain the

moderate performance, particularly with ORG entities. The performance in Afrikaans and Swahili is the lowest, which could be attributed to the morphological complexity and syntactic structures being from different continent. These languages may also have fewer cognates and borrowed words in common with English, making cross-lingual transfer learning more challenging.

Across languages, ORG entities consistently present more significant challenges than LOC and PER entities. This might be due to the variability in organization names, which can include acronyms, non-standard word use, and language-specific naming conventions that are harder for the model to generalize across languages. In contrast, names of locations and people might follow more predictable patterns or have more cognates across languages, aiding in their recognition.

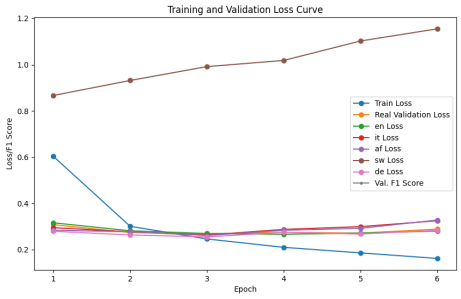

Figure 10: Loss and F1 curve for M-DeBerta v3 trained with English corpus only for evaluating performance.

Language	LOC	ORG	PER	**Macro F1-Score**
English (en)	**0.872**	0.757	0.895	**0.841**
Italian (it)	0.793	**0.777**	**0.934**	0.835
Afrikaans (af)	0.780	0.771	0.884	0.812
Swahili (sw)	0.636	0.455	0.845	0.645
German (de)	0.795	0.682	0.874	0.784

Figure 11: NER Model Performance Across Different Languages for *mdeberta* trained for 4 epoch for Figure 10

1.6 Hyperparameter tuning

We conduct different experiments to analyze the effects of different hyperparameters on the final results.

1.7 Frozen vs Finetuning

First, we use the multilingual BERT model to analyze the effects of fine-tuning the whole model or the classification head only.

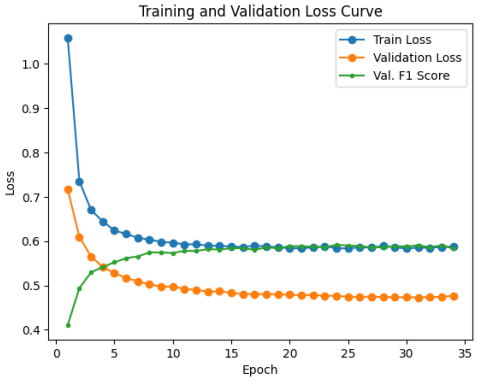

Figure 12: Loss and F1 curve for frozen *bert-base-multilingual-cased* trained for 35 epochs with English corpus, model and setting similar to Figure 8.

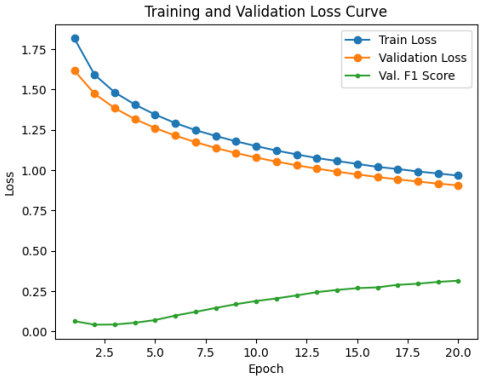

Figure 13: Loss and F1 curve for frozen *M-DeBERTa-v3* trained for 20 epochs with English corpus, model and setting similar to Figure 10.

Model Status	Precision	Recall	F1 Score
Frozen	0.648	0.598	0.622
Non-Frozen	0.837	0.835	0.836

Table 5: Comparison of performance metrics between frozen (Classification Head Only) and non-frozen model (Full Model) of *bert-base-multilingual-cased*. Loss curves on Figure 12.

The results in Table 5 clearly indicate that fine-tuning the entire model (Non-Frozen case) significantly outperforms the approach where only the classification head is trained (Frozen case). This suggests that allowing the model to adjust its pre-trained weights to the specificities of the NER task leads to better model generalization and a higher ability to correctly identify named entities. Also, the model used for training is designed to be fine-tuning to for adapting the pre-trained model to a specific task or domain by updating its parameters with a small amount of labeled data. As the pre-training objective was very different form what we are tying to use the model for.

We further wanted to double-check this results using our best model, M-Deberta v3. The result is reported in Figure 13. Similarly to the previous experiment, the model takes longer to converge. In this case, we decided not to continue with the training to save time and resources.

1.7.1 Varying Learning Rates

We have also chosen to modify the learning rate. We decided to conduct three experiments with different learning rates. We obtain the following as shown in Figure 14, 15 and 16.

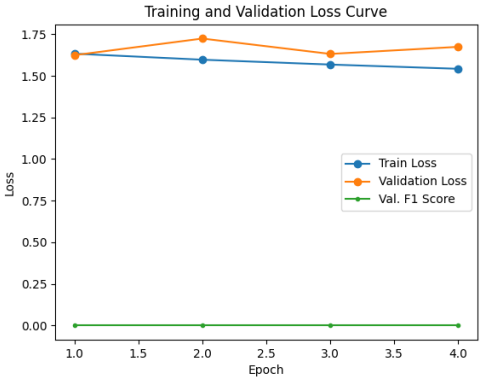

Figure 14: (a) Loss and F1 curve for M-DeBerta trained with English corpus only for evaluating performance with 1e-3 learning rate.

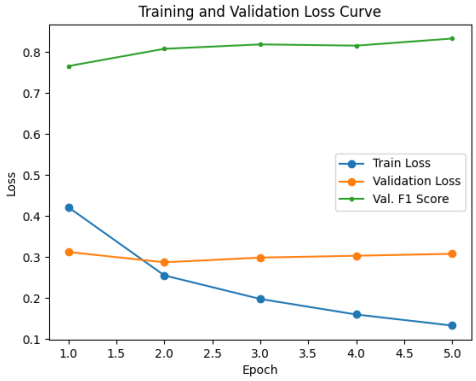

Figure 15: (b) Loss and F1 curve for M-DeBerta trained with English corpus only for evaluating performance with 1e-4 learning rate.

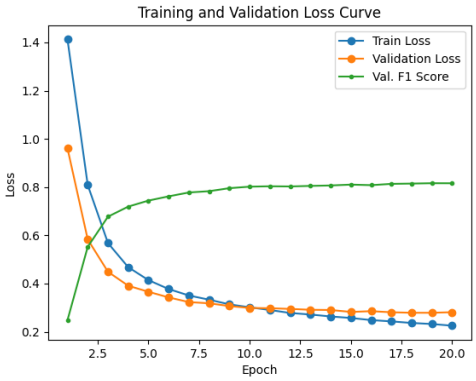

Figure 16: (c) Loss and F1 curve for M-DeBerta trained with English corpus only for evaluating performance with 1e-6 learning rate.

Surprisingly, the biggest learning rate makes the model unable to learn. Bigger learning rates, while potentially decreasing the number of epochs needed to train, can make the model unable to learn. Sometimes, higher learning rates result in models diverging after a certain amount of training. However, most likely due to the model being pretrained, the model is simply unable to learn from the start. The second largest learning rate ($1e-4$) exemplifies this. The model takes fewer epochs to converge. On the other hand, the smaller learning rate obtains a very similar result to the original rate. Nonetheless, it can be seen that extra epochs were taken to achieve the same results. We capped the number of epochs to 20 to preserve resources. We summarize the results of the experiment in Table 6.

Model	Learning Rate	Frozen Layers	Epochs	Macro F1-Score
M-DeBerta v3	1e-5	No	6	0.849
M-DeBerta v3	1e-5	Yes	-	0.376
M-DeBerta v3	1e-3	No	4	0.000
M-DeBerta v3	1e-4	No	7	0.835
M-DeBerta v3	1e-6	No	+20	0.834

Table 6: Results obtained from hyperparameter variation on English dataset.

2 Part 2: Finetuning Multilingual Transformers on Mixture of Languages

2.1 Extending training set (MBERT)

We use *bert-base-multilingual-cased* with same hyper parameters as previous experiments in Section 1.5.

2.1.1 Train Language(s): English+Italian ['en', 'it']

Dataset statistics are shown in Table 8.

Model: *bert-base-multilingual-cased*. Trainable parameters: 177,268,231.

Early stopping at epoch 7 as validation loss has not decreased for 3 epochs since epoch-4.

Table shows validation scores for each language.

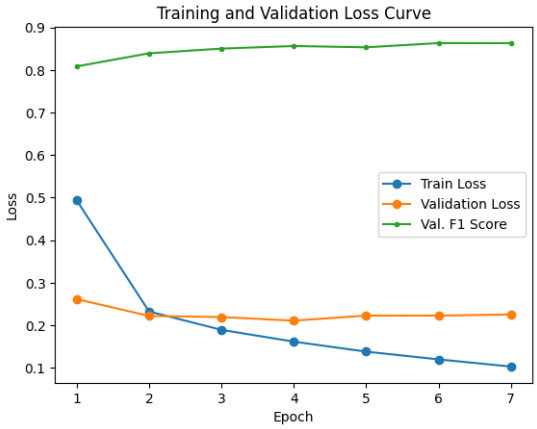

Figure 17: Loss and F1 curve for a model trained with English+Italian corpus. Performance scores in Figure 2.1.1.

Lang	Precision	Recall	F1
en	0.841	0.842	0.841
it	0.909	0.910	0.909
af	0.714	0.796	0.753
sw	0.677	0.670	0.673
de	0.772	0.765	0.768

Table 7: Language-wise Performance Metrics for the model trained on ['en', 'it'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	I-LOC	B-LOC	B-ORG	B-PER
Train	32000	258292	49280	142420	32565	21131	17734	15401	13909	15132
Val	8000	64589	18388	35710	8184	5299	4244	3796	3558	3798
Test(it)	10000	80005	18461	47357	8992	5863	4166	4600	4116	4911
Test(af)	1000	10894	3404	7764	911	548	189	529	583	370
Test(sw)	1000	5609	1830	2412	696	674	618	452	354	403
Test(de)	10000	97805	23570	69057	6107	6539	2284	4968	4281	4569

Table 8: Dataset Statistics for ['en', 'it']

2.1.2 Train Language(s): English+German ['en', 'de']

Dataset statistics are shown in Table 10. Test dataset statistics same as Table 8.
Model: *bert-base-multilingual-cased*. Trainable parameters: 177,268,231.
Early stopping at epoch 7 as validation loss has not decreased for 3 epochs since epoch 4.
Table 9 shows validation scores for each language.

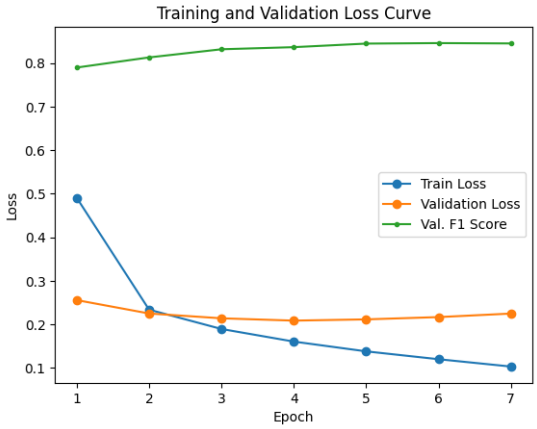

Figure 18: Loss and F1 curve for a model trained with English+German corpus. Performance scores in Figure 9.

Lang	Precision	Recall	F1
en	0.846	0.838	0.842
it	0.799	0.809	0.803
af	0.709	0.804	0.752
sw	0.709	0.643	0.674
de	0.860	0.866	0.862

Table 9: Language-wise Performance Metrics for the model trained on ['en', 'de'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	32000	284404	56417	174868	28295	22468	15269	14763	14415	14326
Val	8000	71377	20823	44029	7177	5560	3854	3691	3395	3671

Table 10: Dataset Statistics for ['en', 'de'].

2.1.3 Train Language(s): English+Italian+German ['en','it,'de']

Dataset statistics are shown in Table 12. Test dataset statistics same as Table 8.
Model: *bert-base-multilingual-cased*. Trainable parameters: 177,268,231.
Early stopping at epoch 7 as validation loss has not decreased for 3 epochs since epoch-4.
Table 11 shows validation scores for each language.

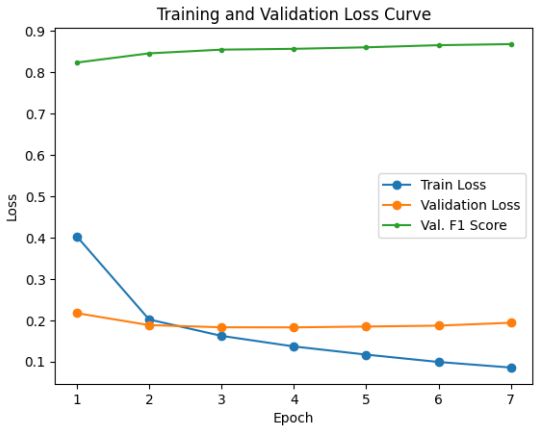

Figure 19: Loss and F1 curve for a model trained with English+Italian+German corpus. Performance scores in Figure 11.

Lang	Precision	Recall	F1
en	0.842	0.846	0.844
it	0.904	0.912	0.908
af	0.763	0.870	0.813
sw	0.670	0.687	0.675
de	0.859	0.874	0.866

Table 11: Language-wise Performance Metrics for the model trained on ['en', 'it', 'de'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	48000	414625	75401	252626	42480	31735	23148	22564	21232	20840
Val	12000	103643	28396	63039	10515	8025	5827	5656	5379	5202

Table 12: Dataset Statistics for ['en', 'it', 'de'].

2.2 Extending training set (DeBerta)

We repeat the experiment above with *DeBerta* model with same hyper parameters as previous experiments in Section 1.5.

2.2.1 Train Language(s): English+Italian ['en', 'it']

Dataset statistics are shown in Table 8.

Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.

Early stopping at epoch 7 as validation loss has not decreased for 3 epochs since epoch-4.

Table shows validation scores for each language.

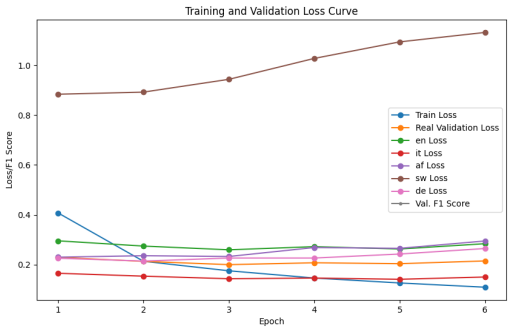

Figure 20: Loss and F1 curve for a model trained with English+Italian corpus. Performance scores in Figure 13.

Lang	Precision	Recall	F1
en	0.852	0.846	0.849
it	0.917	0.918	0.917
af	0.807	0.872	0.838
sw	0.675	0.668	0.669
de	0.664	0.666	0.662

Table 13: Language-wise Performance Metrics for the model trained on ['en', 'it'].

2.2.2 Train Language(s): English+German ['en', 'de']

Dataset statistics are shown in Table 10. Test dataset statistics same as Table 8.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Early stopping at epoch 7 as validation loss has not decreased for 3 epochs since epoch 4.
Table 14 shows validation scores for each language.

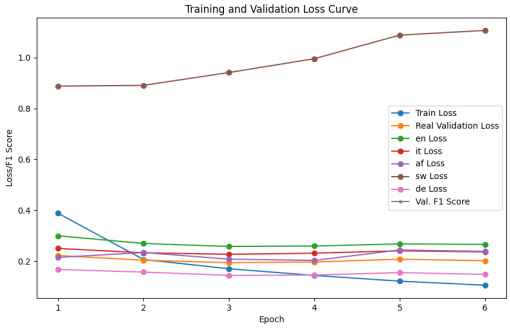

Figure 21: Loss and F1 curve for a model trained with English+German corpus. Performance scores in Figure 14.

Lang	Precision	Recall	F1
en	0.854	0.850	0.851
it	0.869	0.861	0.865
af	0.809	0.877	0.839
sw	0.698	0.660	0.665
de	0.874	0.879	0.876

Table 14: Language-wise Performance Metrics for the model trained on ['en', 'de'].

2.2.3 Train Language(s): English+Italian+German ['en','it,'de']

Dataset statistics are shown in Table 12. Test dataset statistics same as Table 8.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Table 15 shows validation scores for each language.

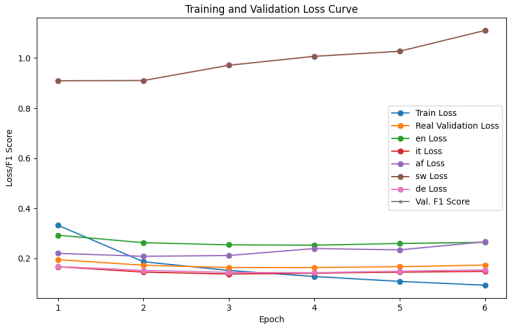

Figure 22: Loss and F1 curve for a model trained with English+Italian+German corpus. Performance scores in Figure 15.

Lang	Precision	Recall	F1
en	0.855	0.854	0.855
it	0.919	0.917	0.918
af	0.810	0.878	0.842
sw	0.704	0.677	0.687
de	0.874	0.879	0.876

Table 15: Language-wise Performance Metrics for the model trained on ['en', 'it', 'de'].

2.3 Selection of extra languages

In the following, we have chosen two additional languages from the provided data set. We continued using the *DeBerta* model with same hyper parameters as previous experiments above. After careful consideration, we have decided to choose Afrikaans and Hungarian. The first choice, Afrikaans, was chosen because it belongs to the Germanic family of languages. This can refine the ability of the model to understand Germanic-derived languages and therefore improve its performance on a wide range of languages at once. Secondly, Hungarian was chosen precisely for the opposite reason. Hungarian belongs to the family of Uralic languages, which represents a significant difference from any of the other languages chosen. In summary, the two extra languages have been chosen to first secure the model's ability to comprehend Germanic languages, and to expand its knowledge to a completely different family of languages to improve its generalizability.

2.3.1 Train Language(s): English+Afrikaans ['en','af']

Dataset statistics are shown in Table 17. Test dataset statistics same as Table 17.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Table 16 shows validation scores for each language.

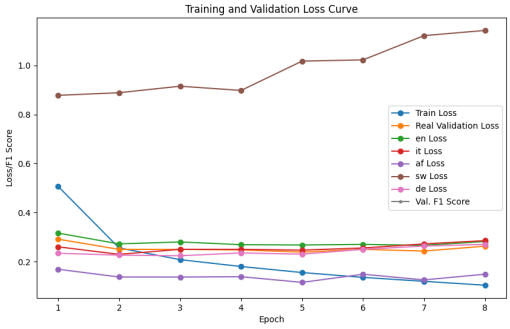

Figure 23: Loss and F1 curve for a model trained with English+Afrikaans corpus. Performance scores in Figure 16.

Lang	Precision	Recall	F1
en	0.854	0.846	0.850
it	0.865	0.853	0.859
af	0.895	0.921	0.907
sw	0.709	0.661	0.668
de	0.813	0.808	0.810

Table 16: Language-wise Performance Metrics for the model trained on ['en', 'af'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	19997	172095	35989	96452	22184	14053	11239	9899	9437	8831
Val	5000	43111	13339	24409	5550	3423	2673	2489	2364	2203

Table 17: Dataset Statistics for ['en', 'af'].

2.3.2 Train Language(s): English+Italian+German ['en','hu']

Dataset statistics are shown in Table 19. Test dataset statistics same as Table 19.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Table 18 shows validation scores for each language.

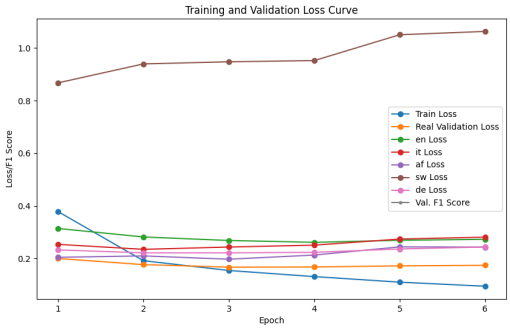

Figure 24: Loss and F1 curve for a model trained with English+Hungarian corpus. Performance scores in Figure 18.

Lang	Precision	Recall	F1
en	0.841	0.846	0.844
it	0.845	0.850	0.847
af	0.814	0.891	0.851
sw	0.671	0.671	0.668
de	0.809	0.823	0.815

Table 18: Language-wise Performance Metrics for the model trained on ['en', 'hu'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	31997	272061	59016	164316	27929	20485	16622	14402	14299	14008
Val	8000	68042	21611	41584	6775	5191	3951	3736	3484	3321

Table 19: Dataset Statistics for ['en', 'hu'].

2.3.3 Train Language(s): English+Italian+German ['en','af','hu']

Dataset statistics are shown in Table 21. Test dataset statistics same as Table 21.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Table 20 shows validation scores for each language.

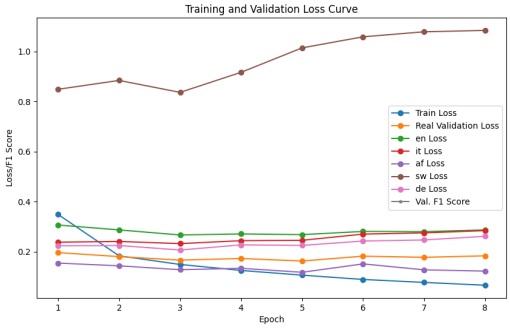

Figure 25: Loss and F1 curve for a model trained with English+Afrikaans+Hungarian corpus. Performance scores in Figure 20.

Lang	Precision	Recall	F1
en	0.855	0.849	0.852
it	0.865	0.856	0.860
af	0.908	0.924	0.916
sw	0.667	0.656	0.658
de	0.836	0.816	0.825

Table 20: Language-wise Performance Metrics for the model trained on ['en', 'af', 'hu'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	35997	316570	66200	196715	31209	22766	18645	16250	15918	15067
Val	9000	79095	24501	48966	8003	5688	4647	4208	3987	3596

Table 21: Dataset Statistics for ['en', 'af', 'hu'].

2.3.4 Train Language(s): English+Italian+German ['en','it','de','af','hu']

Dataset statistics are shown in Table 23. Test dataset statistics same as Table 23.
Model: *mdeberta-v3-base*. Trainable parameters: 278,224,135.
Table 22 shows validation scores for each language.

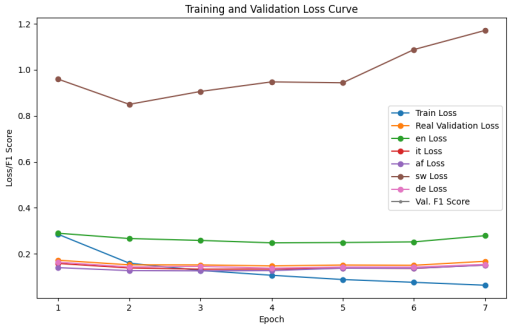

Figure 26: Loss and F1 curve for a model trained with English+Italian+German+Afrikaans+Hungarian corpus. Performance scores in Figure 22.

Lang	Precision	Recall	F1
en	0.860	0.849	0.855
it	0.923	0.919	0.921
af	0.888	0.925	0.906
sw	0.689	0.677	0.679
de	0.888	0.886	0.887

Table 22: Language-wise Performance Metrics for the model trained on ['en', 'it', 'de', 'af', 'hu'].

Set	Length	Tokens	Unique	O	I-ORG	I-PER	B-LOC	B-PER	I-LOC	B-ORG
Train	67998	602876	110309	383812	55407	42888	34343	31273	29612	25541
Val	17000	150662	41511	96172	13574	10628	8579	7688	7466	6555

Table 23: Dataset Statistics for ['en', 'it', 'de', 'af', 'hu'].

2.4 Analysis of results

Given the task of named entity recognition using large language models, adding more languages does not always equal improved performance. The effect of multilingual training data is complex and depends on many factors. For example, the similarity between languages, the quality of data for each language, and the entities wanting to be recognized are only some of said factors. Then we try to describe the benefits and drawbacks of multilingual training.

2.4.1 Benefits

- Model generalization: Training on multiple languages results in a model capable of understanding multiple languages. The trained models will be able to perform better on languages that are part of its training data. Given the resources required to train and fine-tune a model, it is an important benefit to create models that can be used in many situations and not to waste said resources on training extremely fine-tuned models.

- Knowledge transfer: Some languages are closely related to each other. Training data in one language may improve the performance of the model on a different language. For example, Norwegian belongs to the Indo-European Germanic branch. We can assume that training in languages such as German, which usually has more samples, can help increase the performance of the model on Norwegian-related tasks. In our case, we can see that training in Afrikaans improves the performance on German. This is most likely due to the fact that Afrikaans is a Germanic language. It can be particularly important in languages which hold less amount of data such as Afrikaans.

2.4.2 Challenges

- Data imbalance: When it comes to languages, training data is usually heavily skewed towards languages such as English or Spanish. Although it may help for certain languages, it can also pose a problem for the model to learn unrelated languages with smaller datasets. Even if our model was pre-trained in Swahili, the performance is inferior to the other languages. For example, in our experiments, the Swahili language suffered a performance decrease when learning other unrelated languages. Furthermore, when fine-tuning the model on Afrikaans, the ratio of training data to English is lower.

- Linguistic features: Some languages have certain characteristics that can affect the performance of some tasks. In our case, we can assume that languages such as Japanese, which do not have whitespaces to separate words, may be more difficult for our task of named entity recognition. For the experiments, we can see how languages such as Swahili due to the agglutinative nature of the language is a challenge for our task.

- Resource limitations: Although it would be ideal to train with an unlimited amount of data, we should take into account the available resources and the cost. Training and fine-tuning models consume a considerable amount of electricity nowadays. Furthermore, the effort involved in collecting and processing datasets for multiple languages can be substantial. Therefore, many resource-related aspects should be analyzed before training multilingual models.

2.4.3 Best model

We have selected the model fine-tuned with 5 languages as our best model from Section 2.3.4. Although we have seen that fine-tuning on more languages does not necessarily result in better performance, it does help with the generalizability of the model. As previously discussed, many aspects should be taken into consideration, not only the best performance on target metrics. The model chosen performs well on the 5 test sets chosen in this report. Furthermore, the base pre-trained model, M-DeBerta, has been trained in 102 languages. This provides a strong foundation for many other languages. Instead of choosing Swahili as our last additional language, we have decided to choose Hungarian which belongs to the Uralic family of languages. This should provide our model with advantage over other Uralic-derived languages such as Sami or Baltic-Finnic.

Link to Github Repository