AaltoNLP at SemEval-2022 Task 11: Ensembling Task-adaptive Pretrained Transformers for Multilingual Complex NER

Aapo Pietiläinen **Shaoxiong Ji**
Aalto University
{aapo.pietilainen; shaoxiong.ji}@aalto.fi

Abstract

This paper presents the system description of team AaltoNLP for SemEval-2022 shared task 11: MultiCoNER. Transformer-based models have produced high scores on standard Named Entity Recognition (NER) tasks. However, accuracy on complex named entities is still low. Complex and ambiguous named entities have been identified as a major error source in NER tasks. The shared task is about multilingual complex named entity recognition. In this paper, we describe an ensemble approach, which increases accuracy across all tested languages. The system ensembles output from multiple same architecture task-adaptive pretrained transformers trained with different random seeds. We notice a large discrepancy between performance on development and test data. Model selection based on limited development data may not yield optimal results on large test data sets.

1 Introduction

SemEval-2022 shared task 11: MultiCoNER (Malmasi et al., 2022b) was about complex Named Entity Recognition (NER) in the multilingual context. Transformer-based models such as BERT (Devlin et al., 2019) have reached high accuracy in standard NER tasks. However, the models still struggle with complex and code-mixed named entities (Meng et al., 2021; Fetahu et al., 2021). The shared task tries to address the challenges regarding complex NER in a multilingual context. Out of the 13 available tracks, we focus on 5 monolingual tracks. We build models for Bangla, English, Farsi, German, and Korean.

Our strategy is to build an ensemble approach to increase accuracy compared to the baseline model. To tackle multilingualism, we build an approach that starts from the same XLM-RoBERTa (Conneau et al., 2019) encoder and after fine-tuning, achieves good performance across languages. We propose two ensembling approaches: (1) naive ensembling and (2) end-to-end (E2E) ensembling. Naive ensembling is a test-time ensembling where class scores from individually trained models are added together. E2E ensembling trains two encoder models jointly by concatenating their results and passing the concatenated predictions to the final layer. In addition to these strategies, we adapt the encoder model to data context using continued pretraining (Gururangan et al., 2020).

We discover that naive ensembling produces good results with few models and by just using different random seeds for training. We also discover that performance can drastically vary between data sets and model selection based on development data may not yield good results on large test sets. Our final rankings are 11th for Farsi, 12th for Bangla, 13th for German, 15th for Korean, and 25th for English. For each language, our ranking is on the lower half of the participants. The code for our submission has been released [1].

2 Background

In NER, complex and ambiguous entities are hard to classify correctly. Out of the 13 available tracks, we participated in the monolingual tracks for Bangla, English, Farsi, German, and Korean. For training, we used only the competition data sets (Malmasi et al., 2022a) provided by the organizers. The task is to detect named entities in given sentences. Figure 1 contains an example of the task setting. This example can be considered complex, as Madagascar is ambiguous as it could refer to the country or the movie.

Transformer-based models (Vaswani et al., 2017) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) produce state-of-the-art results on variety of natural language processing (NLP) tasks including NER. Transformer models trained

[1]Code is available here https://github.com/aapop/multiconer_AaltoNLP

O	O	O	O	B-LOC	O
it	is	known	from	madagascar	.

Figure 1: An example of NER task. A sentence from the development data set and the corresponding labels.

on multilingual data, such as XLM (Conneau and Lample, 2019), can be used for cross-lingual tasks. XLM-RoBERTa (XLM-R) (Conneau et al., 2019) combines the XLM and RoBERTa architectures and produces state-of-the-art results across languages. XLM-R is used as the basis in our system, which we built on top of.

Gururangan et al. (2020) showed that continuing pretraining of a transformer model with the domain or task-specific data often increases the performance. We utilize this finding and apply task-adaptive pretraining on multilingual models to increase performance in a specific language.

Ensemble learning is widely used in machine learning, in which, predictions from several different models are combined. The ensemble learning method combines multiple weak learners and provides a strong learner (Mohri, 2012). This works especially well when the weak learners [2] are diverse and make different errors. Speck and Ngonga Ngomo (2014) showed that ensemble methods can significantly increase NER accuracy. Combining transformer-based model predictions has been shown to increase NER accuracy (Li et al., 2020; Zhao et al., 2021; Souza et al., 2020). To keep applicability on languages with limited number of pretrained models, our approach is to form an ensemble model based on training the same model with different random seeds.

3 System overview

The motivation for our system is to produce an approach that can be applied across languages. Initial testing with the baseline system provided by the organizers suggested that XLM-R$_{\text{large}}$ (Conneau et al., 2019) produces comparable or even better performance than tested monolingual pretrained models for targeted languages. Therefore, we resort to building on top of a multilingual encoder model XLM-R$_{\text{LARGE}}$ and the provided baseline system.

Inspired by results using task-adaptive pretraining (Gururangan et al., 2020), we start by training task-adaptive language models. We hypothesize that task-adaptive pretraining could provide two sources of improvement: (1) adapting to the data domain and (2) improving the capability in the specific language. Task-adaptive pretraining is conducted for the XLM-R$_{\text{LARGE}}$ using the provided training data, which we prepare for pretraining by removing labels and constructing line-by-line sentences. The pretraining is applied to each language separately yielding us five encoder models named *koala/xlm-roberta-large-**XX***. The **XX** is replaced with the corresponding language identifier. The pretraining is continued from the XLM-R$_{\text{large}}$ checkpoint with only the preprocessed task data. The pretraining objective remains the same. The models are available at the Huggingface model repository [3]. In this paper, we refer to the models as koala-XLM-R$_{\text{large}}$ interchangeably for all languages.

After obtaining the enhanced encoder models, we propose two ensemble learning approaches, i.e., naive ensemble and E2E ensemble.

3.1 Naive ensemble

As the first ensemble strategy, we combine predictions from multiple independently trained models at test time. Randomness and choosing a random seed can have a significant effect on the model training and final accuracy. Based on that, our hypothesis was to leverage this randomness to increase the accuracy.

To ensure applicability in languages, where a limited number of available models are available, we train the models using the same architecture and only vary the random seed associated with training. The koala-XLM-R$_{\text{large}}$ is used as the encoder model for each language respectively. After the encoder model, we use the linear layer as the prediction head and apply softmax transformation to obtain class probabilities. The models were trained separately and optimized with AdamW optimizer (Loshchilov and Hutter, 2018) and negative log-likelihood loss function (NLLLoss).

The ensembling is conducted during test time. We simply add the class probabilities together from each of the models,

$$y^* = \text{argmax}_j \sum_n p_{nj}, \qquad (1)$$

where p_{nj} is the probability of model n for the class j. The predicted class y^* is the class with the highest sum of probabilities. For the ensemble weights,

[2] also called base models

[3] https://huggingface.co/koala

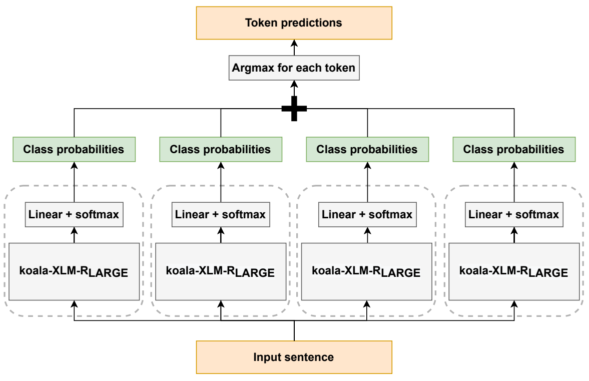

Figure 2: System diagram for the Naive ensemble model. Each of the models are trained separately with different random seeds and ensembling conducted at test time.

we also tried entropy minimized ensemble (Wang et al., 2021), which is based on the intuition that good ensemble weights should decrease entropy (Shannon, 1948; Wang et al., 2020). The algorithm is elaborated in Algorithm 1. This approach did not

Algorithm 1 Entropy minimization for ensembling

```
1: Inputs: scores $\mathbf{p}$, uniform weights $\mathbf{w}^0$
2: for t in $0, 1, \ldots$ steps $= T$ do
3:     Calculate entropy of scores $E \leftarrow \text{Entropy}(\sum_n w_n^t * p_{nj})$
4:     Compute gradient $g^t = \nabla_w E$
5:     Update weights $\mathbf{w}^{t+1} \leftarrow \text{Update}(\mathbf{w}^t, g^t)$
6: end for
7: Output: final prediction $y^* = \sum_n w_n^T * p_{nj}$
```

provide noticeable improvement in our tests and thus, we proceeded with uniform weights.

Our ensemble model consists of four distinct models. The number of four models was chosen as a compromise on the trade-off between accuracy and inference time. The model system is illustrated in Figure 2.

3.2 E2E ensemble

Our second ensemble strategy centered around the idea of jointly training the models. The idea is to include sub-networks (encoder models) in the model, ensemble their predictions, and optimize the loss function jointly. Our system uses koala-XLM-R$_{\text{large}}$ and XLM-R$_{\text{base}}$ as the encoder models.

The input data is passed to both encoder models. After that, we apply linear and softmax layers. For the second encoder model, the linear layer outputs $2 *$ `num classes` scores. We made this decision to add variation between the two sub-networks. The choice to use twice the amount of parameters was arbitrary. We tried a different number of parameters but the performance seemed similar.

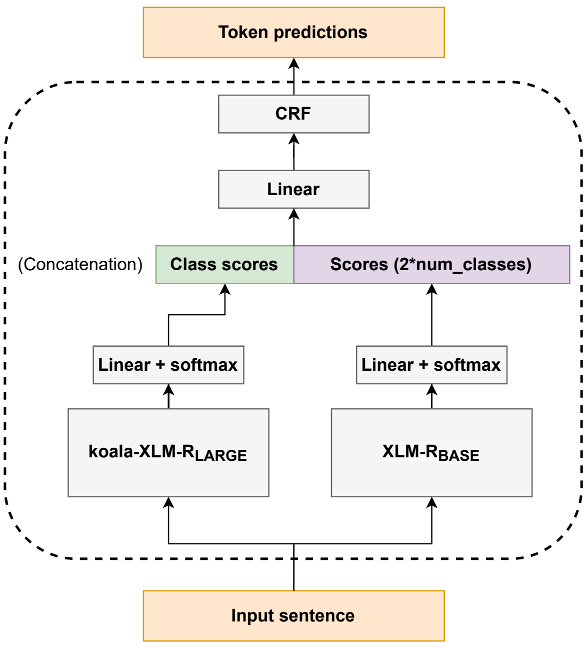

Figure 3: System diagram for the E2E ensemble model.

After outputting the scores from the two sub-networks, the scores are concatenated together. Also, summation was tested when using same-sized outputs, but concatenation seemed to perform better. After that, the concatenated scores are passed to linear and CRF layers. The architecture of E2E ensemble model is illustrated in Figure 3.

The E2E ensemble model consists of two sub-networks. We tried using also three and four sub-networks but adding more models did not improve the performance. Therefore, we settled on using two sub-networks, which also help with the computational burden. We experimented with different sub-network architectures and noticed that using different encoder models provided the best performance despite varying the linear layer size. Also, using koala-XLM-R$_{\text{large}}$ instead of XLM-R$_{\text{base}}$ as the second sub-network did not improve accuracy. Hence, the smaller model was used.

The model was optimized using AdamW optimizer with Viterbi loss as the loss function.

4 Experimental setup

Our experimental setup was based on the provided baseline model (Malmasi et al., 2022b) and the data splits (Malmasi et al., 2022a). Models were trained only using the training data and the best model was selected based on performance on development data. After selecting the best model, we did not do any training with development data. For continued pretraining, we preprocessed the training data. The data were converted from CoNLL format into line-by-line text. Hence, we removed the labels and reconstructed the sentences into a single line of text data.

We selected suitable hyperparameters based on performance on the development data. We selected two learning rates, first for the encoder model and a second one for all subsequent layers (decoder). We set the encoder learning rate to 10^{-6} and the decoder learning rate to 10^{-3}. To prevent overfitting, we used a dropout rate of 0.1. We also used early stopping with development data. The training was stopped when accuracy on development data had not increased in the last three epochs. We selected batch size based on the hardware constraints. For baselines and sub-models in the naive ensemble, we used batch size 64. For E2E ensemble, batch size is 20.

The main Python libraries we used are PyTorch, PyTorch Lightning, AllenNLP, and Transformers. Further details available in the repository [4].

The performance in this shared task is evaluated using prediction accuracy on unseen test data set. The teams are ranked by their macro-averaged F1 classification score.

5 Results

The official metrics were based on the evaluation phase with test data. The participants were ranked by their Macro F1 scores. The performance of built models on test data is shown in Table 1. The best score is bolded and is the model used for ranking. It can be seen that the naive ensemble model performs better than the best individual model across all languages. E2E ensemble model shows mixed results and performance varies run by run. It outperforms naive ensemble on Korean and Farsi test data.

We noticed a rather large discrepancy in accuracies between the development and test data set. The details of model performance on development data are presented in Table 2. Compared to the results on test data, the performance on development data was significantly better. The discrepancies between performance on development and test data suggest that model selection on such limited development data yields sub-optimal results.

[4] https://github.com/aapop/multiconer_AaltoNLP/blob/master/requirements.txt

Model	en	ko	fa	de	bn
XLM-R$_{large}$	0.639	0.589	0.515	0.695	0.465
koala-XLM-R$_{large}$	0.653	0.590	0.527	0.695	0.476
Naive ensemble	**0.669**	0.610	0.569	**0.714**	**0.518**
E2E ensemble	0.623	**0.618**	**0.589**	0.678	0.511
Final rank	25	15	11	13	12

Table 1: Macro F1 scores and final ranking on test data.

Model	en	ko	fa	de	bn
XLM-R$_{large}$	0.830	0.802	0.756	0.870	0.763
koala-XLM-R$_{large}$	0.829	0.805	0.745	0.861	0.758
Naive ensemble	**0.841**	**0.824**	**0.764**	**0.864**	**0.782**
E2E ensemble	0.821	0.811	**0.764**	0.856	0.755

Table 2: Macro F1 scores on development data.

Our systems suffer with some of the classes. As can be seen from Figures 4 and 5, PROD and CW classes have significantly lower accuracy than other classes. Performance on GRP, LOC, and PER classes is much higher. Our systems suffer with the no-tag-class O, which has the majority of the misclassifications.

5.1 Task-adaptive pretraining

We hypothesized that task-adaptive pretraining would adapt the multilingual encoder model into the data and language domains. We tested the performance by training the model four times using different random seeds and comparing the average micro F1 score between XLM-R$_{large}$ and koala-XLM-R$_{large}$ models. The results of task-adaptive pretraining are elaborated in Table 3. The improvements on this task are small and compared to the original paper, in which, the accuracy increase was considerably larger on some data sets. For German, the accuracy even decreased significantly. The decrease is mainly caused by random variation as one of the four models for koala-XLM-R$_{large}$ achieved only accuracy of 0.81 when all other models (XLM-R$_{large}$ and koala-XLM-R$_{large}$) were in the range of $0.85 - 0.88$.

Model	en	ko	fa	de	bn
XLM-R$_{large}$	0.836	0.788	0.757	0.873	0.738
koala-XLM-R$_{large}$	0.837	0.793	0.761	0.850	0.746
Improvement	0.1%	0.6%	0.5%	-2.6%	1.1%

Table 3: Average micro F1 scores for four models. Models trained with different random seeds.

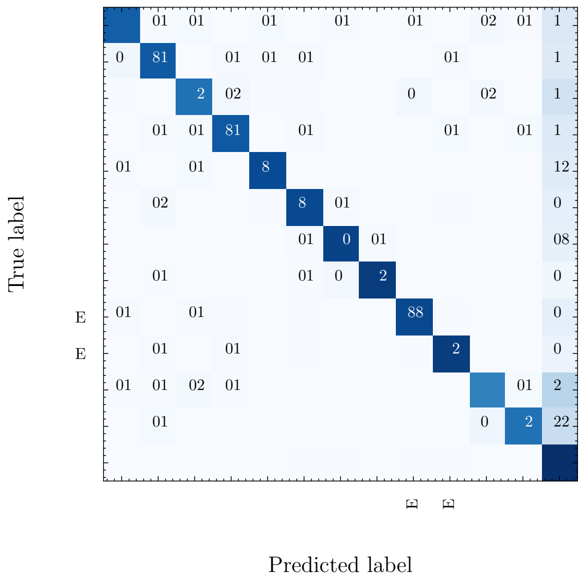
Figure 4: Confusion matrix of Farsi Naive ensemble model normalized on the true labels.

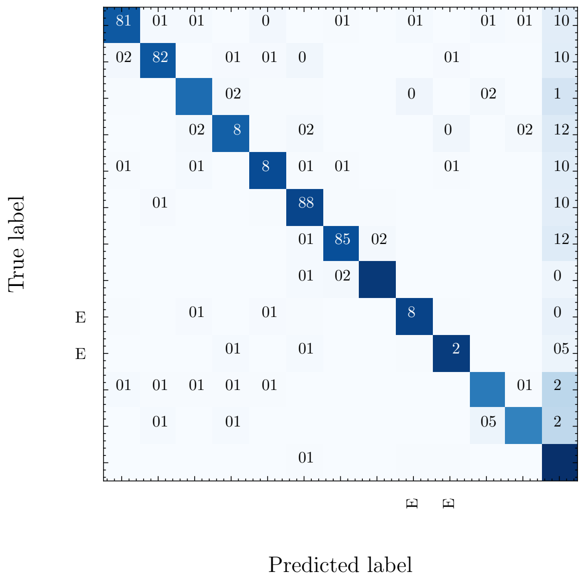
Figure 5: Confusion matrix of Farsi E2E ensemble model normalized on the true labels.

EMEA	Steps	Lr	Batch size	F1
No	-	-	-	**0.782**
Yes	10	10	3	0.779
Yes	10	100	3	0.776
Yes	10	500	3	0.777
Yes	25	10	1	0.771
Yes	30	15	3	0.775
Yes	100	10	3	0.777

Table 4: EMEA ensemble for Bangla

5.2 EMEA

As discussed earlier, we tried to improve our ensembling strategy using entropy minimization. The results are reported in Table 4. Despite testing different approaches, no sign of improvement is present. The testing was conducted on the Bangla development data set. Our setting differs from the original authors as we are not using off-the-shelf language adapters. We probably have too few models and they are not diverse enough.

6 Conclusion

From our efforts, we conclude that naive ensembling improves accuracy with just four models of the same architecture trained with different random seeds. The ensemble of four models outperforms the best individual model across all the tested languages. The E2E ensemble model can provide good accuracy, but the results vary drastically between runs. Task-adaptive pretraining, which has in some cases improved accuracy significantly, yielded only a slight improvement in this task. We also notice a large difference between performance in development and test data. With such limited development data and a large test set, model selection on solely development data yields sub-optimal results. More attention should be paid to model selection and model's generalizability.

Acknowledgments

We thank the Aalto Science-IT project and CSC - IT Center for Science, Finland, for providing the computational resources, that were necessary to participate in this task. We would like to thank the shared task organizers for the excellent arrangements. We thank Feiyi Wang for the initial discussion.

References

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. *CoRR*, abs/1911.02116.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. *Advances in neural information processing systems*, 32.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Besnik Fetahu, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. Gazetteer Enhanced Named Entity Recognition for Code-Mixed Web Queries. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1677–1681.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks.

Xiangyang Li, Huan Zhang, and Xiao-Hua Zhou. 2020. Chinese clinical named entity recognition with variant neural structures based on bert methods. *Journal of biomedical informatics*, 107:103422.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022a. Multiconer: a large-scale multilingual dataset for complex named entity recognition.

Shervin Malmasi, Anjie Fang, Besnik Fetahu, Sudipta Kar, and Oleg Rokhlenko. 2022b. Semeval-2022 task 11: Multilingual complex named entity recognition (multiconer). In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Tao Meng, Anjie Fang, Oleg Rokhlenko, and Shervin Malmasi. 2021. GEMNET: Effective gated gazetteer representations for recognizing complex entities in low-context input. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1499–1512.

Mehryar Mohri. 2012. Foundations of machine learning.

Claude Elwood Shannon. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. 2020. Bertimbau: pretrained bert models for brazilian portuguese. In *Brazilian Conference on Intelligent Systems*, pages 403–417. Springer.

René Speck and Axel-Cyrille Ngonga Ngomo. 2014. Ensemble learning for named entity recognition. In *International semantic web conference*, pages 519–534. Springer.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. 2020. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*.

Xinyi Wang, Yulia Tsvetkov, Sebastian Ruder, and Graham Neubig. 2021. Efficient Test Time Adapter Ensembling for Low-resource Language Varieties.

Lingyun Zhao, Lin Li, Xinhao Zheng, and Jianwei Zhang. 2021. A bert based sentiment analysis and key entity detection approach for online financial texts. In *2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 1233–1238. IEEE.