HeLju@VarDial 2020: Social Media Variety Geolocation with BERT Models

Yves Scherrer
Department of Digital Humanities
University of Helsinki
yves.scherrer@helsinki.fi

Nikola Ljubešić
Department of Knowledge Technologies
Jožef Stefan Institute
nikola.ljubesic@ijs.si

Abstract

This paper describes the Helsinki–Ljubljana contribution to the VarDial shared task on social media variety geolocation. Our solutions are based on the BERT Transformer models, the constrained versions of our models reaching 1st place in two subtasks and 3rd place in one subtask, while our unconstrained models outperform all the constrained systems by a large margin. We show in our analyses that Transformer-based models outperform traditional models by far, and that improvements obtained by pre-training models on large quantities of (mostly standard) text are significant, but not drastic, with single-language models also outperforming multilingual models. Our manual analysis shows that two types of signals are the most crucial for a (mis)prediction: named entities and dialectal features, both of which are handled well by our models.

1 Introduction

Until 2019, all VarDial evaluation campaigns have focused on classification tasks where the number of linguistic varieties was defined beforehand by the task organizers. The 2020 SMG task breaks with this tradition and asks the participants to predict latitude-longitude coordinates, i.e., two output values on a continuous scale. The SMG task is divided in three subtasks focusing on different linguistic areas: the Bosnian-Croatian-Montenegrin-Serbian (BCMS) language area, the German-speaking area of Germany and Austria (DEAT), and German-speaking Switzerland (CH). All three datasets are based on social media data, Twitter in the case of BCMS and Jodel in the case of DEAT and CH.

This paper describes the HeLju (Helsinki–Ljubljana) submission to the SMG task. Our motivation was to investigate how existing classification and regression approaches can be adapted to a double regression task. Most of our work is based on the BERT sentence classification architecture in both constrained and unconstrained settings. We experiment with various pre-trained models, different types of coordinate encoding and other hyperparameters. Finally, we manually analyze the development set predictions made with our best-performing models.

2 Related work

One of the first works focusing on predicting geolocation from social media text is Han et al. (2012). The authors investigate feature (token) selection methods for location prediction, showing that traditional predictive algorithms yield significantly better results if feature selection is performed.

There has been already a shared task on geolocation prediction at WNUT 2016 (Han et al., 2016). The task focused not only on predicting geolocation from text, but also from various user metadata. The best performing systems were combining the available information via feedforward networks or ensembles.

Thomas and Hennig (2018) report significant improvements over the winner of the WNUT-16 shared task by learning separately text and metadata embeddings via different neural network architectures

	BCMS	DEAT	CH
Training instances	320 042	336 983	22 600
Development instances	39 750	46 582	3 068
Test instances	39 723	48 239	3 097
Median instance length	12	45	47
Maximum instance length	44	272	129

Table 1: Data characteristics. Instance length statistics are computed on space-separated token count of the training sets.

(LSTM, feedforward), merging those embeddings and performing the final classification via a softmax layer.

To the best of our knowledge, large pre-trained language models such as BERT have not yet been applied to the problem of geolocation prediction, but just to language identification (Bernier-Colborne et al., 2019).

3 Data

The VarDial evaluation campaign provides training, development and test data for the three subtasks. Table 1 gives an overview of the data. It can be seen that the BCMS and DEAT datasets are roughly equivalent in size, whereas the CH dataset is more than one order of magnitude smaller. The instances from the Twitter dataset (BCMS) correspond to single tweets, while the instances from the Jodel datasets correspond to entire conversations and are thus much longer on average.

The task organizers also provide a simple baseline for the geolocation task. They compute the centroid ("average location") of all instances in the training data and then predict the coordinates of this centroid for all development or test instances (see Table 2, first row).

4 Experiments

The geolocation task takes text as input and produces two real-valued outputs, the predicted latitude and longitude. The two outputs can be produced by independently trained models or by a single model that benefits from some form of parameter sharing.

4.1 Traditional machine learning approaches

In dialect classification, one of the most popular and successful approaches relies on an SVM classifier that is trained on character n-grams. It has also been shown beneficial to weight the n-gram frequencies, e.g. using the TF-IDF weighting scheme (Jauhiainen et al., 2019; Martinc and Pollak, 2019). We adapted this setup by substituting the SVM classifier by two independent SVR regression models, one for latitude and one for longitude.[1] As input features, we used TF-IDF-weighted n-grams of length 3 to 6 occurring at least 5 times in the training corpus. For this and all other experiments presented in the paper, we train and test our systems on lower-cased data, as we found no evidence that casing information would be relevant for geolocation. No further pre-processing was applied to the data. Table 2 (second row) shows that this approach easily beats the centroid baseline for all three subtasks.

4.2 Neural machine learning approaches

In recent years, pre-trained language representations have become very successful for various down-stream tasks. BERT (Devlin et al., 2019) is currently one of the most popular pre-trained language representation frameworks and is based on the Transformer neural network architecture (Vaswani et al., 2017). Typically, a BERT model is created in two phases. In the pre-training phase, a Transformer is trained from scratch using a masked language modeling task. This task only requires unlabeled data. A

[1]We used the *MultiOutputRegressor* class of the Scikit-Learn toolkit (Pedregosa et al., 2011) to combine the two models.

	Median distance (km)		
Model	BCMS	DEAT	CH
Centroid baseline	107.1	201.3	41.4
SVR with TF-IDF character n-grams	82.2	168.7	29.7
Constrained BERT	48.1	159.2	17.8
Multilingual BERT	44.9	150.7	16.6
Language-specific BERT	42.7	146.5	15.6

Table 2: Median distances (lower is better) of different models on the development set.

wide variety of pre-trained models have been made publicly available. Since the Transformer architecture requires a fixed-size vocabulary, each pre-trained model also comes with a pre-trained tokenizer that splits infrequent words into subword units. In the fine-tuning phase, a pre-trained model is adapted to a particular downstream task. For example, in an instance classification task, the output representation of the special `[CLS]` token is fed into a separate fully-connected layer that predicts the output class. In this phase, training data with gold class labels are required. Bernier-Colborne et al. (2019) have successfully used BERT for dialect identification. The fine-tuning step can be adapted easily from a classification to a regression problem by removing the sigmoid function and choosing an appropriate loss function. Likewise, double regression can be implemented by producing two-dimensional output vectors.[2]

We report three preliminary experiments with the BERT architecture here.[3] In the first experiment, we pre-train a BERT model from scratch on the VarDial SMG training data (without the coordinates),[4] and then fine-tune it on the regression task with the SMG training data (including the coordinates). Just like the SVR setup, this is a constrained setup in the sense that no data sources other than the ones provided by VarDial are used. The third row in Table 2 shows that BERT outperforms the SVR massively on all three tasks despite using exactly the same data. In particular, the median distances are reduced by more than 40% for the BCMS and CH tasks.

In the second experiment, we use the pre-trained `bert-base-multilingual-uncased` model[5] and fine-tune it on the VarDial data; we use the provided tokenizer without any adaptation. This model has been pre-trained on about 100 languages, including German, Croatian and Serbian. However, we do not expect it to be particularly well adapted to our task for two reasons. First, massively multilingual models are prone to capacity dilution: at constant model capacity, the part allocated to each language is inversely proportional to the number of languages covered by the model (Conneau et al., 2020). Second, its tokenizer is trained on all the languages it supports, yielding therefore suboptimal text splitting in comparison to language-specific tokenizers. Despite these shortcomings, the distances (Table 2, fourth row) are further reduced by 5–7% compared to the constrained setup. Note that in preliminary experiments, we also tested the XLM-RoBERTa model (Conneau et al., 2020), but found no benefits compared with the original multilingual BERT.

For the third experiment, we looked for alternative pre-trained models that overcome the limitations of multilingual BERT and specifically cover the languages of the three subtasks. For BCMS, we use the `crosloengual-bert` model provided by Embeddia (Ulčar and Robnik-Šikonja, 2020).[6] For DEAT, we use the `bert-base-german-dbmdz-uncased` model provided by the Bavarian State Library.[7] As we were not able to find a pre-trained model for Swiss German, we started with the German BERT

[2]For our experiments, we adapt the *simpletransformers* library (`https://simpletransformers.ai/`), which already supports single regression fine-tuning. This library is built on top of the HuggingFace *Transformers* library (Wolf et al., 2019).

[3]For all BERT experiments reported in this section, we use the hyperparameters specified in Sections 4.3 and 4.4, but report numbers from single runs.

[4]We pre-trained the models for 50 000 steps with a batch size of 32, for all three models. For each model, a WordPiece tokenizer with vocabulary size 30 000 is trained on the same data.

[5]`https://huggingface.co/bert-base-multilingual-uncased`

[6]`https://huggingface.co/EMBEDDIA/crosloengual-bert`

[7]`https://huggingface.co/dbmdz/bert-base-german-uncased`

and continued pre-training for another five epochs on the SwissCrawl corpus (Linder et al., 2020). For all three tasks, we rely on the provided tokenizers without modifications. This third setup further improves geolocation results (see last line of Table 2).

Our final submissions are based on the first BERT setup for the constrained setting, and on the third BERT setup for the unconstrained setting.

4.3 Hyperparameter tuning

During our initial experiments, we found the BERT models to be quite sensitive to some hyperparameter settings. We found that the optimal batch sizes for the fine-tuning step depended on the amount of available training data. We obtained the best results with batch sizes of 64 for BCMS, 128 for DEAT, and 32 for CH.

Although the pre-trained BERT models support maximum sequence lengths of up to 512 tokens, we found a smaller maximum length of 128 tokens to work equally well for the BCMS and CH task, while for DEAT the optimal maximum sequence length was 256 tokens, both of which reduced the GPU memory requirements.[8] The *simpletransformers* library also provides a sliding window option that splits long instances into pieces instead of just cutting them off at the maximum length. We did not see any beneficial effect of this option, which suggests that the beginning of each conversation is most indicative of its geographic localization.[9]

During fine-tuning, intermediate models were saved every 2000 training steps and the savepoint with the lowest median distance value measured on the development set was selected. The best models were generally obtained around epoch 8 for BCMS, epoch 4 for DEAT, and epoch 50 for CH. All models tended to overfit to the training data thereafter.

4.4 Coordinate encoding and loss functions

Fine-tuning a BERT model on a regression task requires the selection of an appropriate loss function. This loss function should ideally correspond to the evaluation measure used in the task. The official evaluation measure is median distance measured with the Haversine formula (which assumes that the Earth is a perfect sphere with a radius of 6371 km). While it is possible to train models using the Haversine loss function, we found it more promising to apply some conversions to the coordinates and to use standard regression loss functions instead. We explored various configurations:

Coordinate projections The raw latitude and longitude coordinates have some properties that make them potentially hard to learn: (1) one latitude degree does not amount to the same number of kilometers than one longitude degree; (2) one longitude (East-West) degree corresponds to a larger number of kilometers near the Equator than near the poles. We propose three alternative projections that can be used in conjunction with standard loss functions:

- **Cosine-adjusted longitude:** All longitude values are multiplied by the cosine of their corresponding latitude. The latitude values remain unchanged.[10]
- **UTM projection:** The latitude-longitude coordinates are converted to the UTM coordinate system. This system is based on rectangular zones where each point is represented as the metric distance from the origin of the zone. We chose zone 34T for BCMS, 32U for DEAT, and 32T for CH.[11]
- **Cartesian coordinate system:** The latitude-longitude coordinates are converted into a triple of X, Y and Z coordinates in a 3-dimensional space where the origin corresponds to the center of the Earth.

[8]Note that this sequence length is computed after tokenization. While the median instance length before tokenization lies between 12 and 47 (see Table 1), the median instance length after tokenization varies between 24 and 78. The chosen threshold lies thus still well above the median.

[9]This may also be an artifact of the collection method for DEAT and CH, where each conversation is assigned the coordinate of its first message.

[10]See `https://stackoverflow.com/a/1664836`.

[11]See `https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_system`.

		Average median distances (↓) BCMS	DEAT	CH	Average relative distance reduction to baseline (↑)
Projection	Lat-Lon	49.86	**149.30**	15.92	46.9%
	Cos-adj. Lat	50.57	150.80	15.93	46.5%
	UTM	49.61	149.95	**15.86**	**47.0%**
	Cartesian	**49.41**	149.63	15.90	**47.0%**
Loss function	MSE	56.36	151.84	16.04	44.4%
	MAE	**43.36**	**148.00**	**15.76**	**49.3%**
Scaling	Independent	**49.72**	150.18	**15.87**	**46.9%**
	Joint	50.00	**149.65**	15.93	46.8%

Table 3: Results of parameter search experiments. The rightmost column represents averages over the three tasks in the form of relative distance reduction percentages, i.e. higher values are better.

Loss functions We experiment with two commonly used regression loss functions, mean absolute error (MAE/L1) and mean square error (MSE/L2). The former corresponds to Manhattan distance, whereas the latter corresponds to Euclidean distance. While both loss functions apply equally well to multi-dimensional regression, we expect Euclidean distance to be more appropriate for geographic space.

Loss reduction Commonly, the individual losses of a batch are reduced to a single value by taking their mean or sum. Since the official evaluation measure is based on the median, we tried a median reduction as well, but with consistently poor results. We therefore continue to use the mean reduction.

Centering and scaling Neural networks tend not to converge well if the output values are large or not well distributed. A common solution for this problem consists in standardizing the values by centering (i.e., subtracting the mean of the training data) and scaling (i.e., dividing by the standard deviation of the training data). Since our data is two-dimensional, standardizing the values of both dimensions independently may lead to distortion, as (assuming both value distributions are normal) the coordinates are being forced into a square even though the original space is not square-shaped. To prevent this, we propose joint standardization, where the mean is computed independently for each dimension, but the standard deviation is computed jointly on all (mean-removed) values.

Leaving aside the median reductions, which underperformed in preliminary experiments, we trained a total of 48 models (3 tasks $\times$ 4 projections $\times$ 2 loss functions $\times$ 2 scalings) and evaluated each on the respective development set.

The first three columns of Table 3 show average median distances taken over all models of a task, keeping one of the parameters fixed. The last column shows an average over the three tasks; in order to make the numbers comparable, we convert the median distances to distance reduction percentages relative to the task-specific baseline and report the mean reduction percentages over the three tasks.

In terms of projection, we find that the cosine-adjusted latitude encoding performs worse than the other three, even though it satisfies the same requirements as UTM, for example. Surprisingly, raw latitude and longitude do not perform worse than the metric coordinate systems. In terms of loss function, the results show a clear and consistent advantage for MAE loss. We currently do not have an explanation for this result, as the MSE loss should capture geographic distances better.[12] Finally, the two scaling methods are indistinguishable. We also tested for interactions between parameters using a mixed-effects regression model, but did not obtain any statistically significant effects. For our final models (both constrained and unconstrained), we opted for raw latitude-longitude, MAE loss and joint scaling.

These findings seem to contradict several intuitions of geographic modelling: (1) some amount of distortion in the coordinate space does not seem to be an issue, since distortion-free projections such as

[12]Note that the same advantage for MAE obtains when evaluated on mean distance instead of median distance.

			Median distance (km)		
System name	Constrained	Rankings	BCMS	DEAT	CH
HeLju 1	no	1,2,2	41.54	143.85	15.72
HeLju 2	no	2,1,1	41.61	143.30	15.45
HeLju 3	yes	1,1,3	48.99	159.59	17.97

Table 4: Final results of our submitted three runs in median distances (lower is better).

UTM or Cartesian do not perform better than raw latitude-longitude, and (2) Manhattan distance seems more appropriate than Euclidean distance. We conclude from this that the geolocation task does not require a geographically faithful modelling of the output space. This suggests that the models do not infer the dialect landscape as a continuum that spreads in two dimensions, but rather as a set of distinct zones (presumably urban areas) whose exact geographic relationships to each other are to some extent arbitrary. We will come back to this hypothesis in Section 5.

4.5 Test-time adaptations

It is well known that neural-network-based models can be sensitive to weight initialization. Therefore, we trained four identical models per task with different random seeds and selected the one with the lowest development set distance for producing the test output.

Furthermore, given that the development sets are rather generous in size, we hypothesized that the information contained therein might be more useful for training than for validation. Therefore, we produced an alternative data split in which the development set is reduced to 3000 (BCMS), 6000 (DEAT) and 1000 (CH) instances respectively, and the remainder is added to the training set. We again train four models per task with this extended data split.

4.6 Results

We submitted three systems per track: an unconstrained system with the default data split (*HeLju 1*), an unconstrained system with the extended data split (*HeLju 2*), and a constrained system with the default data split (*HeLju 3*).

The results of our three submitted systems, with their ranking among all submitted systems, are given in Table 4. Our first observation is that the overall results are very close to the results obtained on dev data, which shows that the dev and test data are quite probably coming from the same distribution. The two unconstrained systems (the only two of their kind among the submitted systems) outperform the constrained version significantly, with the two unconstrained systems performing very similarly, showing that extending the training data with parts of the dev data did not give the extra push that we were hoping for.

The constrained system obtained best results in two out of three subtasks, yielding only third place in the CH subtask that had one order of magnitude less training data than the two other tasks, which also gives the most probable reason for its not-superior performance. In the two tasks where our constrained system won, we we outperformed the second ranked system by an absolute difference in median distance of 8.25 kilometers for the BCMS subtask and 24.4 kilometers for the DEAT subtask. In the CH subtask, the winning system outperformed ours with an absolute difference of 2.04 kilometers.

5 Error analysis

Figure 1 shows the spatial distribution of the development set instances of the three tasks. The plots on the left show the predicted locations, with the color indicating the distance to the true locations. In all three plots, high-accuracy zones can be distinguished (colored in red), which correspond to urban areas. Furthermore, instances that are hard to classify are put into a “default” zone close to the centroid location, in order to minimize distances.

The plots on the right show the true locations, with the color indicating the distance to the predicted

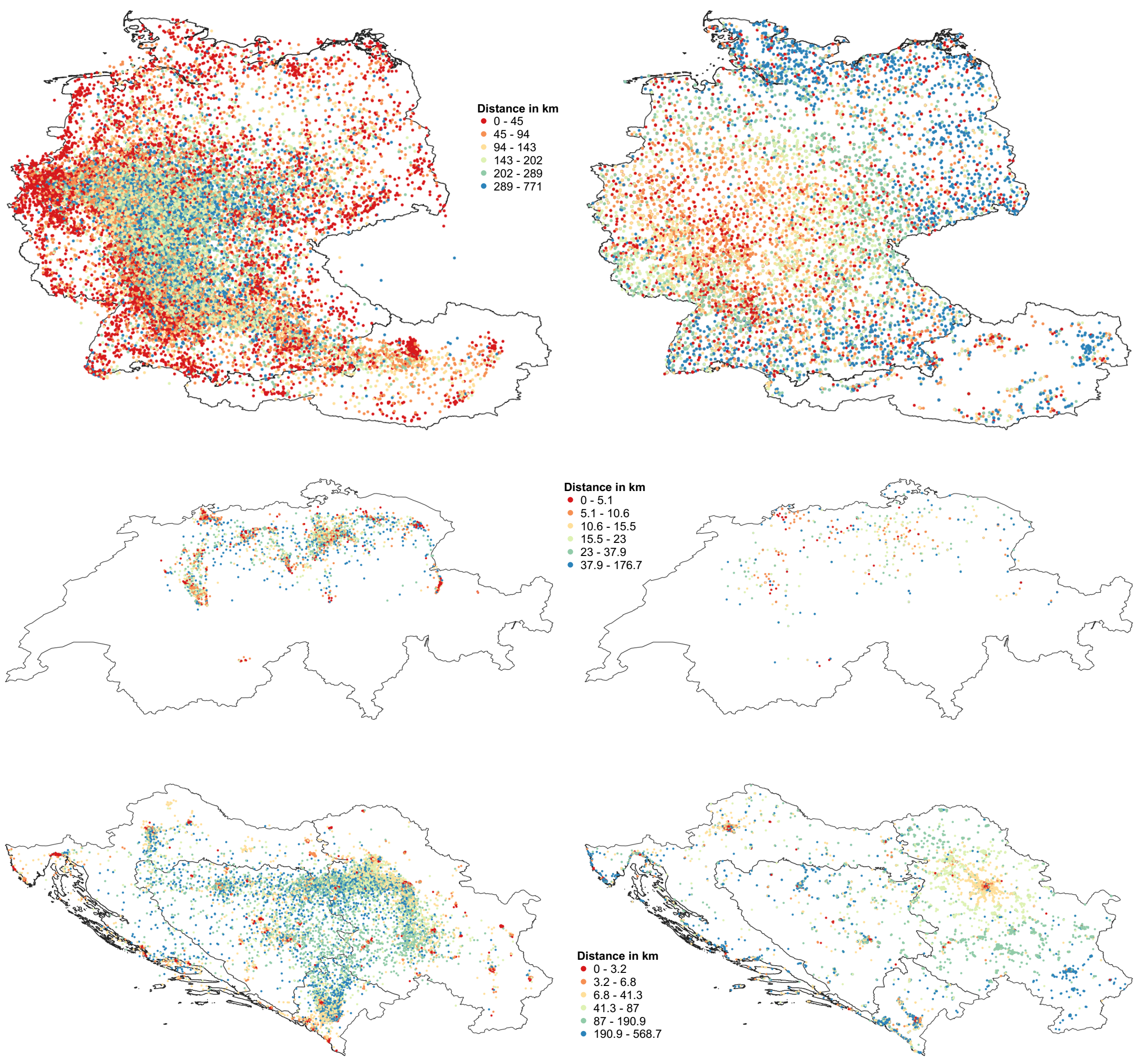

Figure 1: Visualization of development set predictions. Left: instances are placed at their predicted place, the color shows the distance to the true place. Right: instances are placed at their true place, the color shows the distance to the predicted place. Red color stands for small distances, blue color for large distances.

locations. For the DEAT task, red and blue dots occur simultaneously in the same areas. For BCMS, the major urban areas can again be distinguished easily.

We manually analyzed a subset of development set instances for each task. To this end, we chose two areas per task that can be delimited easily and that lie outside of the “default area”. We then selected at most 50 “good” and 50 “bad” predictions, on the basis of a task-specific threshold set according to a natural break occurring in the data. We then counted how many of these instances contain dialectal features and named entities (mostly place names, but also names of well known locations, people, institutions or businesses), and whether these features were used consistently (e.g. an instance located in Berlin mentions *Berlin*) or inconsistently (e.g. an instance located in Berlin mentions *Köln*). Table 5 lists some consistent and inconsistent dialectal features, whereas Table 6 shows the results of this analysis.

This small-scale analysis shows some striking differences between the tasks. For the CH task, good prediction performance is mainly associated with the presence of consistent dialectal features, whereas for the DEAT task, prediction almost exclusively relies on the presence of consistent named entities. The BCMS task lies between those two, with both named entity and dialectal signal being present in good

Task	Region	Consistent features	Inconsistent features
BCMS	Zagreb	*Kolindin ili Josipovicev fejk osmijeh*	*Cao cao #svenasvetu* *da uvik slikaju iste pičine*
	Belgrade	*letnji pljusak je okej* *Ustala sam posle 3 meseca pre 4 sata*	*volio bih da ti kazem* *Upravo sam saznao ovu pretužnu vijest*
CH	Wallis	*Und wier chännes miterläbu!* *ich sägu minum botsch ästagsch öi h.* *Sust nu jemand moru en Statisterolla?*	*Hahaha je confirme y a trop . . .* *Hans liebte Franz doch Franz . . .* *Da isch en Aargaueri[n] uf de . . .*
	Chur	*leid tuats miar glich abitz* *Wär muos am 3.7 nach Airolo irucka?* *Liabi Lüüt fahrend doch eifach 50*	*Bisch demfau z frouefäud?* *Wulewu kuschee awek mua?* *wo konn men do heint fortgian?*
DEAT	Berlin	*dat* *nüscht*	*isch*
	Vienna	*spoats eichs bitte ma jt zu sagen* *Also is bei mir doch nicht so fad* *eine ur schirche kartoffelnase*	–

Table 5: Examples of consistent and inconsistent dialect features.

Task	Region	Criterion	N	Named entities		Dialectal features	
				Consistent	Inconsistent	Consistent	Inconsistent
BCMS	Zagreb	$<$ 20 km	50	54%	0%	40%	4%
		$>$ 100 km	50	4%	10%	0%	16%
	Belgrade	$<$ 20 km	50	24%	0%	48%	0%
		$>$ 100 km	50	4%	4%	0%	20%
CH	Wallis	$<$ 20 km	12	8%	0%	100%	0%
		$>$ 50 km	11	27%	9%	73%	27%
	Chur	$<$ 50 km	106	23%	6%	99%	5%
		$>$ 70 km	6	17%	17%	67%	50%
DEAT	Berlin/	$<$ 70 km	47	94%	11%	2%	0%
	Potsdam	$>$ 400 km	50	8%	10%	4%	2%
	Vienna	$<$ 70 km	13	92%	8%	23%	0%
		$>$ 600 km	50	0%	6%	4%	0%

Table 6: Presence of named entities and dialectal features in well and badly classified subsets of the development set.

predictions.

Bad performance is hard to predict for CH, but seems to be related to inconsistent usage of dialectal features. For DEAT, bad performance seems to be due almost exclusively to the absence of named entities. In the DEAT dataset, dialectal and regional features – consistent or not – are extremely rare. In the BCMS dataset bad prediction performance is mostly to be followed back to lack of useful named entity and dialectal signal, as well as a significant amount of misleading dialectal signal, which is to be followed back to the fact that the two selected regions are the two largest cities in the whole area that are surely visited or inhabited by speakers from around the area.

6 Conclusion

In this paper we have presented the first use of the popular Transformer-based BERT systems on geolocation prediction that ensured a very strong first place in the unconstrained version of the VarDial 2020 SMG shared task and the first place in two out of three subtasks in the constrained version of the shared task.

We have shown that pre-trained models perform drastically better than the usual traditional machine learning approaches, even if both are based on the same significant amount of data available. Models pre-trained on large amounts of text do outperform those pre-trained on the training data only, but not with a drastic difference. This lack of a larger difference might probably be followed back to the lack of dialectal features in the large pre-training data.

Language-specific pre-trained models do outperform multilingual models, but not as drastically as one would expect. This lack of a larger difference can probably again be followed back to the lack of dialectal features in the pre-training data. Finally, performing various target transformations from the original (latitude, longitude) space does not improve results.

Our error analysis has shown that the three subtasks are quite different regarding the useful signal available. While the DEAT decisions mostly rely on named entities, and the CH decisions on dialectal features, the BCMS subtask lies somewhere in the middle, with both signals having similar importance. By having such a diverse test bed, we believe that we have shown that the current state-of-the art natural language models produce also the best results on this task, regardless of the type of the useful signal, as long as a significant amount of data (around million instances) is available for pre-training and fine-tuning.

Acknowledgements

This work has been supported by the FoTran project, funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 771113), and the Slovenian Research Agency through research core funding No. P6-0411 "Language resources and technologies for Slovene language" and the research project ARRS N6-0099 and FWO G070619N "The linguistic landscape of hate speech on social media".

References

Gabriel Bernier-Colborne, Cyril Goutte, and Serge Léger. 2019. Improving cuneiform language identification with BERT. In *Proceedings of the Sixth Workshop on NLP for Similar Languages, Varieties and Dialects*, pages 17–25, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July. Association for Computational Linguistics.

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, June. Association for Computational Linguistics.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Geolocation prediction in social media data by finding location indicative words. In *Proceedings of COLING 2012*, pages 1045–1062, Mumbai, India, December. The COLING 2012 Organizing Committee.

Bo Han, Afshin Rahimi, Leon Derczynski, and Timothy Baldwin. 2016. Twitter geolocation prediction shared task of the 2016 workshop on noisy user-generated text. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 213–217, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Tommi Jauhiainen, Marco Lui, Marcos Zampieri, Timothy Baldwin, and Krister Lindén. 2019. Automatic language identification in texts: A survey. *Journal of Artificial Intelligence Research*, 65:675–782.

Lucy Linder, Michael Jungo, Jean Hennebert, Claudiu Cristian Musat, and Andreas Fischer. 2020. Automatic creation of text corpora for low-resource languages from the internet: The case of Swiss German. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 2706–2711, Marseille, France, May. European Language Resources Association.

Matej Martinc and Senja Pollak. 2019. Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, 25(5):607–632.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Philippe Thomas and Leonhard Hennig. 2018. Twitter geolocation prediction using neural networks. In Georg Rehm and Thierry Declerck, editors, *Language Technologies for the Challenges of the Digital Age*, pages 248–255, Cham. Springer International Publishing.

Matej Ulčar and Marko Robnik-Šikonja. 2020. FinEst BERT and CroSloEngual BERT: less is more in multilingual models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.