AGILE: GIScience Series, 2, 39, 2021. https://doi.org/10.5194/agile-giss-2-39-2021
Proceedings of the 24th AGILE Conference on Geographic Information Science, 2021.
Editors: Panagiotis Partsinevelos, Phaedon Kyriakidis, and Marinos Kavouras.
This contribution underwent peer review based on a full paper submission.

Detecting OpenStreetMap missing buildings by transferring pre-trained deep neural networks

Jan Pisl[a](corresponding author), Hao Li[b], Sven Lautenbach[a], Benjamin Herfort[a,b], Alexander Zipf[a,b]

janjanpisl@gmail.com, hao.li@uni-heidelberg.de, herfort@uni-heidelberg.de, sven.lautenbach@uni-heidelberg.de, zipf@uni-heidelberg.de

[a]HeiGIT at Heidelberg University, Schloss-Wolfsbrunnenweg 33, 69118 Heidelberg, Germany
[b]GIScience Chair, Institute of Geography, Heidelberg University, 69120 Heidelberg, Germany

Abstract.

Accurate and complete geographic data of human settlements is crucial for effective emergency response, humanitarian aid and sustainable development. OpenStreetMap (OSM) can serve as a valuable source of this data. As there are still many areas missing in OSM, deep neural networks have been trained to detect such areas from satellite imagery. However, in regions where little or no training data is available, training networks is problematic. In this study, we proposed a method of transferring a building detection model, which was previously trained in an area well-mapped in OSM, to remote data-scarce areas. The transferring was achieved via fine-tuning the model on limited training samples from the original training area and the target area. We validated the method by transferring deep neural networks trained in Tanzania to a site in Cameroon with straight distance of over 2600 km, and tested multiple variants of the proposed method. Finally, we applied the fine-tuned model to detect 1192 buildings missing OSM in a selected area in Cameroon. The results showed that the proposed method led to a significant improvement in f1-score with as little as 30 training examples from the target area. This is a crucial quality of the proposed method as it allows to fine-tune models to regions where OSM data is scarce.

Keywords. OpenStreetMap, deep learning, building detection, humanitarian mapping, DeepVGI

1 Introduction

Spatial data are fundamental for long-term planning and sustainable development as well as immediate emergency and disaster response. Information on the spatial distribution of human settlements is of particular importance. "Disaster mapping" has been proven as a powerful method to leverage the "ability of volunteers to assist in disaster response situations via mapping an other spatial analysis" (Albuquerque et al., 2016; Herfort et al., 2021), especially in Sub-Saharan Africa, a region struggling with extreme poverty and other humanitarian issues. Still, the availability of such data from local governmental sources remains relatively poor (Humanitarian OpenStreetMap Team, 2019).

The crowdsourced mapping platform OpenStreetMap (OSM) can be a valuable source of this data, especially when data from other sources is not available. Since many areas have still not been mapped sufficiently in OSM, identifying these areas is an important and necessary step towards better reliability of OSM and effective management of volunteer contributions.

Supervised or unsupervised machine learning approaches have been used in a few recent studies for such machine-assisted mapping tasks (Chen and Zipf, 2017). Herfort et al. (2019) proposed a method of training deep neural networks to detect buildings from satellite imagery and assist another crowdsourced mapping platform, namely MapSwipe, to better and faster identify human settlements which are missing in OSM. Li et al. (2020) applied a clustering algorithm of geotagged tweets to identify areas with high potential of human activities. In a second step, a similar approach to (Li et al., 2019) was applied, using pre-trained deep networks to classify image tiles with high probability of OSM missing built-up areas. Both studies trained their networks in areas where OSM coverage of buildings was relatively high and applied their networks in relative proximity to where the networks were trained.

In this study, we focus on enabling deep neural networks to be adapted to remote geographic regions, which aims to identify missing OSM buildings in regions where there is little or no relevant data available in OSM.

The hypothesis is that a trained neural network, namely a model, can perform well in areas with similar appearances to the training region. We used the geographical distance as a proxy for this similarity. It is then assumed that the performance of the model decreases significantly when applied in areas that differ from the training regions. These differences include diverse landscapes, sources of satellite imagery or appearances of buildings.

2 Data and Methods

2.1 Data

Labels acquired from OSM were used to train the model on building detection from Bing satellite image tiles at zoom level of 18. The spatial resolution of the imagery was approximately 0.6 meters. Each image tile corresponded to one training examples and had a size of 256×256 pixels.

Table 1. Overview of data sets.

Name	Country	Type	Buildings
Igombe	Tanzania	Training	14781
Kalola	Tanzania	Validation	340
Bandjoun	Cameroon	Training	4091
Ndem	Cameroon	Validation	1100
Bamendjou	Cameroon	Test	286
Batcham	Cameroon	Test	370
Djeve	Cameroon	Test	359

A total of seven data sets were used. They are detailed on Table 1 and visualized on Figures 1 and 2.

Traning data set Igombe was used to train the base model. It was collected from an area near the city Igombe, Tanzania. This location was selected as it offers a relatively good quality of OSM data considering the finished humanitarian mapping projects in the area.

Training area Bandjoun was used to sample additional training examples to generate fine-tuning data sets. The size of this data set was chosen to test how the base model can be fine-tuned to a data-scarce region.

Validation sets Kalola and Ndem (one in close geographic proximity to each of the two training data sets) were used to compare how different models perform. Test sets Batcham, Djeve and Bamendjou were used to assess the performance of a final selected model on unseen data.

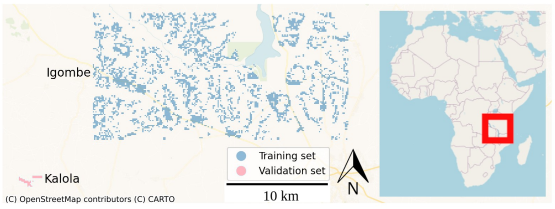

Figure 1. Overview of data sets in Tanzania.

Building geometries in all validation and test sets were manually checked, corrected and completed. Since OSM data is considered the ground truth, when computing performance metric, this manual check ensured accurate statistics regarding model performance will be obtained.

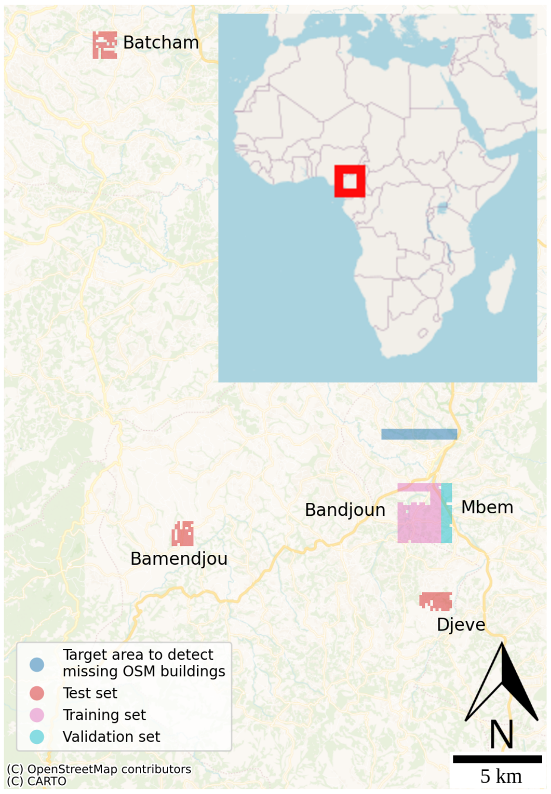

Figure 2. Overview of data sets in Cameroon.

2.2 Methods

This study presents a novel method of adapting (fine-tuning) a previously trained convolutional neural network (the base model) to new locations. The base model was obtained by tuning an SSD Inception V2 network (Szegedy et al., 2016), pre-trained on Microsoft COCO data set (Lin et al., 2015), on a spatially homogeneous data set Igombe, Tanzania. Our method predicted bounding boxes of every individual detected building and we can therefore provide fine-grained information on the volume of buildings present and missing in OSM.

SSD Inception V2 consists of three main parts. First, the backbone network, Inception V2, is responsible for extracting features from the raw input image. The backbone network is truncated before classification layers. Next, additional convolutional layers are added that progressively decrease in size. Each of these layers produces feature predictions. Known as multi-scale feature maps, they allow detection of features at different scales. Finally, classification layers at the end are used to classify predictions.

To calculate loss, the predicted bounding boxes are classified as positive or negative. A prediction is classified as positive if the Intersection over Union (IoU) of the ground truth box and the predicted bounding box is equal or greater than a specified threshold. In this study, an IoU threshold of 0.3 was used. Otherwise, the prediction is classified as negative.

The loss function used for model training is computed as a combination of localization and classification loss (Liu et al., 2016):

$$L(x,c,l,g) = \frac{1}{N}(L_{class}(x,c) + L_{loc}(x,l,g)) \qquad (1)$$

where N is the amount of matched default boxes, $x_{ij}^p = 1,0$ indicates if the i-th default box is a positive match for the j-th ground truth bounding box of class p, c is the correct class, g is the ground truth box and l is the predicted box. Only positive predictions are considered in the localization loss. For details on how localization and classification losses are computed, please refer to Liu et al. (2016).

For training, the RMSprop (root mean squared propagation) algorithm, based on gradient descent and proposed by (Hinton, 2012), was used as an optimizer. Batch size of 24 and an exponential decay learning rate that decreases progressively during training were also used.

The fine-tuning method was inspired by few shot learning, an emerging topic in machine learning characterized by a limited size of training data set. In particular, our inspiration comes from the approach of Wang et al. (2020) who used a two-stage approach: after training the model on base classes C_b, feature extractor data weights were fixed and only the classifier was fine-tuned with an additional, fine-tuning data set. Wang et al. (2020) generate the fine-tuning data set by combining examples of C_b with examples of new-added classes C_n, such that the data set contains an equal number of examples per class.

In this study, the focus was on detecting only a single class - buildings. Nevertheless, there are similarities with the situation of few shot learning. The base model was trained on satellite imagery labeled by OSM data in an area near the city of Igombe, Tanzania (Figure 1), where OSM data is abundant. As per our hypothesis, this model's performance would decrease when used in regions with different appearance from the training data. Because OSM data is scarce in many regions of Sub-Saharan Africa, it is not possible to train a new model for each area in which we want to detect buildings. Therefore, the situation resembles the settings presented in Wang et al. (2020), only instead of a limited number of training examples of new classes, we have a limited number of examples of buildings in a new area.

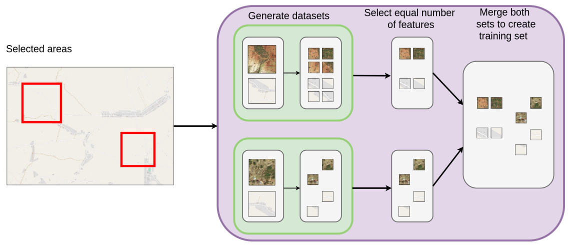
Selected areas
Generate datasets
Select equal number of features
Merge both sets to create training set

Figure 3. The process of generating a training data set for fine-tuning.

We attempted to fine-tune our base model, trained on data set Igombe, Tanzania, to a selected site in Cameroon. Following the methodology of Wang et al. (2020), we generated fine-tuning data sets by combining an equal number of examples from training area Igombe, Tanzania and examples from the data-scarce region Bandjoun, Cameroon.

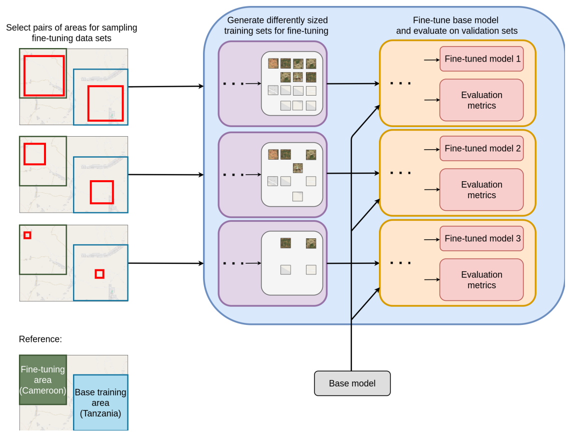
Select pairs of areas for sampling fine-tuning data sets
Generate differently sized training sets for fine-tuning
Fine-tune base model and evaluate on validation sets
Fine-tuned model 1
Evaluation metrics
Fine-tuned model 2
Evaluation metrics
Fine-tuned model 3
Evaluation metrics
Base model
Reference:
Fine-tuning area (Cameroon)
Base training area (Tanzania)

Figure 4. Workflow for comparing multiple fine-tuning variants; the base model is tuned multiple times with different amounts of fine-tuning; Generating a fine-tuning data set is shown on Figure 3.

2.3 Experimental design

A total of six experiments was carried out (cf. table 2). The experiments varied i) the size of the additional training data set for adapting to the new region and ii) which parts of the network were allowed to vary in the adaptation. For the later, two variants were tested and compared: in addition to fixing parameters in the feature extractor and fine-tuning only the classifier, as proposed in Wang et al. (2020), we also tuned all parameters of the network (feature extractor weights not fixed).

The amount of data needed from the new area to which the base model is being tuned was assumed to be crucial as well. Therefore, three different sizes of the fine-tuning data set were tested: the size of the fine-tuning data sets were 590, 200 and 60 with half of the examples from the base data set Igombe, Tanzania, which then means the number of additional examples from the Bandjoun, Cameroon was 295, 100 and 30, respectively. The individual steps of how differently sized tuning data sets are generated, used and evaluated, is shown on Figure 4. Note that the process shown on the figure was executed twice - once tuning only the classifier and once tuning the entire network as explained in previous paragraph. As a result, we obtained six independent fine-tuned models. t

Table 2. Overview of fine-tuning experiments.

Model number	Fine-tuning data set size	Feature extractor weights fixed
1	30+30	False
2	100+100	False
3	295+295	False
4	30+30	True
5	100+100	True
6	295+295	True

Finally, the six models and the base model were compared and the model with the best performance (highest average f1-score, computed over the two validation sets) was selected as the final tuned model, which was then evaluated on three test sets and applied to a selected area in Cameroon where buildings were missing in OSM (shown on Figure 2).

2.4 Software and Data Availability

The Research data and code supporting this publication is available at https://doi.org/10.6084/m9.figshare.14466288. For implementations, we used Python 3.6 (Van Rossum and Drake, 2009) and Tensorflow 1.14 (Abadi et al., 2015) with the Tensorflow Object Detection API [1]. Moreover, the pre-trained parameter for the building detection model was downloaded from the Tensorflow Object Detection API Model Zoo (Huang et al., 2016). Data includes OSM building geometries ($building = *$), and the Bing satellite imagery were collected at zoom level 18 with via the Bing Maps developer API [2].

[1]https://github.com/tensorflow/models

3 Results

3.1 Comparison of fine-tuning variants

Applying the base model to the validation set Ndem, Cameroon resulted in very poor results, specifically in terms of recall and f1-score (cf. table 3). This aligns with our hypothesis that a model's performance decreases when used on data with different appearance than the model's training data. All six adjusted models performed much better, highlighting the value of fine-tuning models with additional local training data. As for performance on validation set Kalola, Tanzania, located nearby where the base model was trained, here the base model without any fine-tuning reached the highest f1-score. However, differences in f1-score were relatively small for this region.

When comparing results achieved by the six fine-tuning variants, it shows that higher performance was achieved when all model parameters were allowed to be tuned during the adaptation training (models 1-3 in Table 2). For those three models, performance increased further with the size of the additional training samples. However, differences in f1-score were relatively small, indicating that even a small additional training set improved model performance substantially compared to the application of the unadjusted base model.

As for the training progress, training models with additional training samples resulted in an immediate strong increase in performance in the new, data-scarce region in Cameroon (cf. apendix, Figures 6 and 7). For the data-rich region Kalola, performance was quite stable over the training period or showed even a slight decrease in f1-score with increased training period for the small number of additional samples if all parameters of the network were allowed to vary. This suggests only little training is required for the observed improvement.

3.2 Evaluation of the final transferred model

Based on the results presented in Table 3, model 3 (specified in Table 2), was selected as the best performing variant of the fine-tuning experiments and evaluated further on three test sets. The results are shown

[2]https://www.bingmapsportal.com/

Table 3. Performance of all models on two validation sets. The base model represents the model that was trained on training data set Igombe, Tanzania, and applied to two validation sets without further adjustments. Models 1-6 represent models that were fine-tuned to Cameroon (cf. table 2). Precision, recall and f1-score are reported for both validation sets.

	Kalola, Tanzania			Mbem, Cameroon		
Model	Prec.	Rec.	F1	Prec.	Rec.	F1
Base	**0.913**	0.865	**0.889**	**0.966**	0.045	0.086
1	0.881	0.810	0.844	0.934	0.752	0.833
2	0.869	0.830	0.848	0.922	0.776	0.842
3	0.859	**0.887**	0.873	**0.946**	0.786	**0.859**
4	0.842	0.879	0.860	0.909	0.721	0.804
5	0.848	0.874	0.861	0.939	0.682	0.791
6	0.840	0.879	0.859	0.896	0.730	0.805

in Table 4. Interestingly, f1-score values the model achieved on all three test sets are higher than on the validation set.

Also, it is notable that the model performed similarly on all three test sets. The location of test sets was intentionally selected so that each has a different distance from the data set Bandjoun that was used to sample training examples for fine-tuning the base model (as seen on Figure 2. Per our hypothesis stated in the introduction, the performance should decrease with increasing distance from the training area. For the three tests, however, this is not the case. This finding might help to answer an important question: when a model is fine-tuned - as in this study - with training examples from Bandjoun, how far from Bandjoun can we use this tuned model before performance starts to decrease? To answer, a more detailed analysis needs to be carried out, but the results in Table 3 show that applying the fine-tuned model at distances of approximately 5, 15 and 30 kilometers (Djeve, Bamendjou and Batcham, respectively) yield comparable results.

Table 4. Performance of a selected fine-tuned variant (model 3) evaluated on three test sets.

Test area	Precision	Recall	F1-score
Batcham	0.943	0.886	0.914
Djeve	0.938	0.884	0.910
Bamendjou	0.937	0.872	0.903

A visual inspection of a test area in Batcham, Cameroon (c.f. Figure 5) shows that the model successfully detected the majority of buildings. Two most common sources of errors might be observed in the figure: first, buildings with relatively small footprints often remain undetected. On the upper image in Figure 5, the only two undetected buildings are significantly smaller than the rest. This behavior was observed with all tested models. Second, buildings located on the border of two or more image tiles were problematic. The detector is designed to detect buildings on a single image tile at a time. If multiple tiles contained parts of a building, the building was often detected multiple times. Although the image tiles' borders are not visible, the two false positives (detections visualized with red color) on the lower image of Figure 5 are examples of this situation.

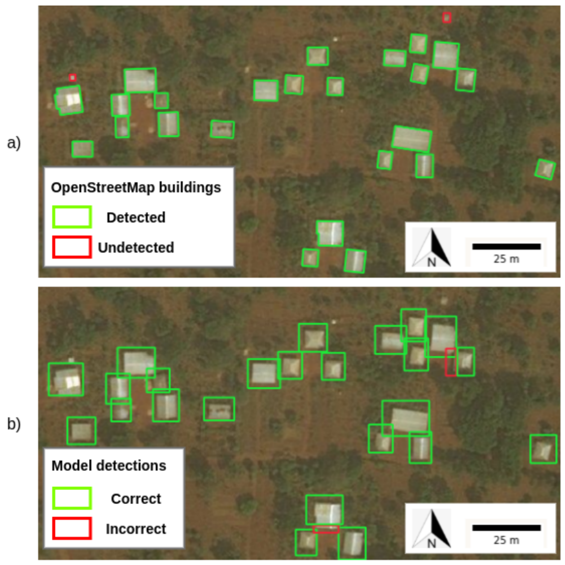

Figure 5. Visualization of results produced by model 3 in a selected area in test area Batcham, Cameroon.

3.3 Missing OSM buildings

To demonstrate the capability of detecting missing OSM buildings, the final selected model (model 3) was then applied to a location where OSM building data were missing completely (see Figure 2). With an area of around 2.58 km^2, the model detected 1192 buildings which were missing in OSM.

4 Conclusion

In this study, we proposed a novel method of detecting buildings missing in OSM by transferring deep neural networks to geographically remote regions with a limited amount of additional training data. We tested multiple variants of the method and demonstrated the capability of the proposed method by adopting a network trained in Tanzania to a site in Cameroon. The building detection model fine-tuned with 295 additional examples achieved an f1-score over 0.9 on all three test sets in Cameroon. Finally, We used the fine-tuned model to detect 1192 buildings missing in OSM within a selected area of approx. 2.5 km^2 in Cameroon.

The proposed model can detect individual buildings by generating bounding boxes around each of them. However, further volunteer contributions are still expected with regard to mapping building footprints as well as validation. Therefore, this work contributes to the efforts towards machine-assisted humanitarian mapping methods. Future works will focus on better facilitating and supporting the OSM mapping community by estimating amounts of missing buildings or prioritizing unmapped areas.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, https://www.tensorflow.org/, 2015.

Albuquerque, J. P. d., Herfort, B., and Eckle, M.: The Tasks of the Crowd: A Typology of Tasks in Geographic Information Crowdsourcing and a Case Study in Humanitarian Mapping, Remote Sensing, 8, https://doi.org/10.3390/rs8100859, https://www.mdpi.com/2072-4292/8/10/859, 2016.

Chen, J. and Zipf, A.: DeepVGI: Deep learning with volunteered geographic information, in: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 771–772, 2017.

Herfort, B., Li, H., Fendrich, S., Lautenbach, S., and Zipf, A.: Mapping Human Settlements with Higher Accuracy and Less Volunteer Efforts by Combining Crowdsourcing and Deep Learning, Remote Sensing, 11, https://doi.org/10.3390/rs11151799, www.mdpi.com/journal/remotesensing, 2019.

Herfort, B., Lautenbach, S., de Albuquerque, J. P., Anderson, J., and Zipf, A.: The evolution of humanitarian mapping within the OpenStreetMap community, Scientific reports, 11, 1–15, 2021.

Hinton, G.: Neural Networks for Machine Learning Lecture 6a Overview of mini-batch gradient descent, http://www.cs.toronto.edu/~hinton/coursera/lecture6/lec6.pdf, 2012.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., and Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors, CoRR, abs/1611.10012, http://arxiv.org/abs/1611.10012, 2016.

Humanitarian OpenStreetMap Team: Mapping 2021: HOT's Strategic Plan 2019 – 2021, Tech. rep., https://www.hotosm.org/strategic-plan, 2019.

Li, H., Herfort, B., and Zipf, A.: Estimating OpenStreetMap missing built-up areas using pre-trained deep neural networks, in: Proceedings of the 22nd AGILE Conference on Geographic Information Science, pp. 17–20, 2019.

Li, H., Herfort, B., Huang, W., Zia, M., and Zipf, A.: Exploration of OpenStreetMap missing built-up areas using twitter hierarchical clustering and deep learning in Mozambique, ISPRS Journal of Photogrammetry and Remote Sensing, 166, 41–51, https://www.sciencedirect.com/science/article/pii/S0924271620301271, 2020.

Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P.: Microsoft COCO: Common Objects in Context, 2015.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A.: SSD: Single Shot MultiBox Detector, vol. 9905, pp. 21–37, https://doi.org/10.1007/978-3-319-46448-0_2, 2016.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z.: Rethinking the Inception Architecture for Computer Vision, in: Computer Vision and Pattern Recognition 2016, https://doi.org/10.1109/CVPR.2016.308, 2016.

Van Rossum, G. and Drake, F. L.: Python 3 Reference Manual, CreateSpace, Scotts Valley, CA, https://doi.org/https://dl.acm.org/doi/book/10.5555/1593511, 2009.

Wang, X., Huang, T. E., Darrell, T., Gonzalez, J. E., and Yu, F.: Frustratingly simple few-shot object detection, arXiv preprint arXiv:2003.06957, 2020.

Appendix

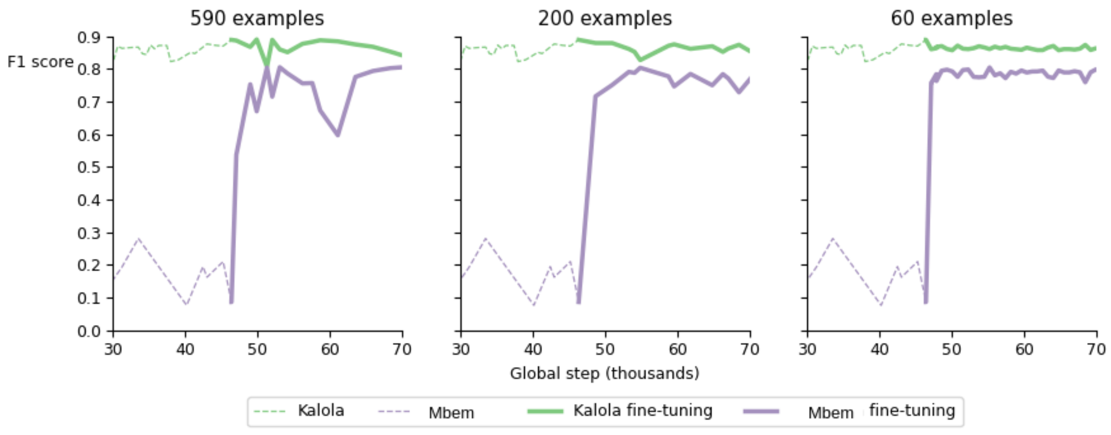

Figure 6. Training progress on the two validation sets - only tuning the classifier.

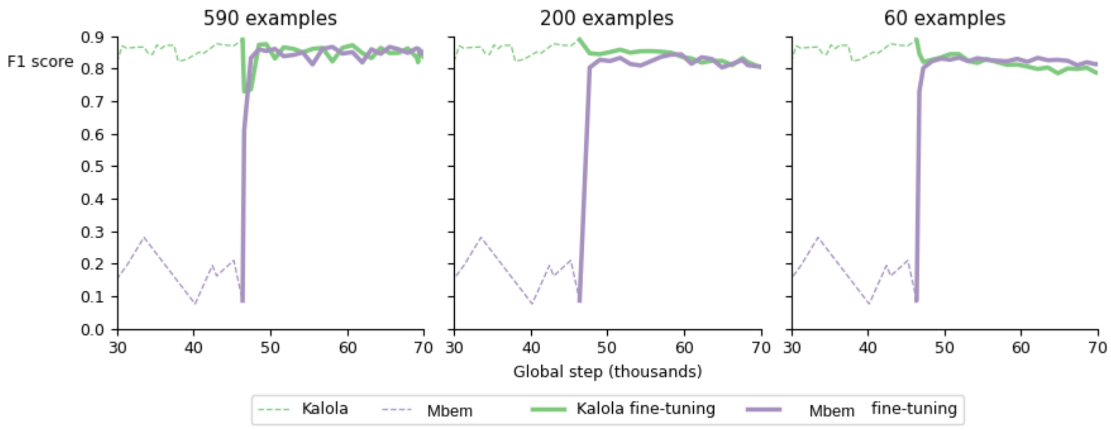

Figure 7. Training progress on the validation sets - tuning all parameters in the network.