Reproduction and Exploration of Group-Instance Cost Function Model

Andi Dai
andi.dai@mail.mcgill.ca

Yiran Mao
yiran.mao@mail.mcgill.ca

Jingyuan Wang
jingyuan_wang@outlook.com

September 10, 2019

Abstract

In this project we reproduced and modified the algorithm proposed by the paper "From Group to Individual Labels using Deep Features" [1]. In the paper the author focused on the multi-instance learning with sparse label problem, where we have labels for groups of instances but not for the instances themselves. To solve this, the author intended to learn classifiers at the instance level and the key idea was to propose a new cost function that inferred sentence implication label on the basis of the instance-level similarity, while keeping group-level label constraints simultaneously. In our reproducing implementation, we used two large review datasets from IMDb and Amazon and embedded at sentence level to evaluate the algorithm. Furthermore, we investigated the dependency of results on the training hyperparameters and did ablation studies on the proposed function itself. The ablation studies demonstrated the importance of both group cost and instance similarity cost, though the kernel function in the similarity cost was replaceable in a large extent. Overall, our work proved the proposed approach was both accurate and scalable in multi-instance classification task.

1 Introduction

In many classification problems labels are relatively scarce. That is, the label is given to a group of instances rather than the instances itself. For example, one movie review may contain several sentences, i.e. instances, but only one label. Past work on this problem has typically focused on learning classifiers to make predictions at the group level. However, it is rational to solve a multi-instance classification problem[2] at the instance level since each sentence can be viewed as sub-documents and better labeling in sub-document units will leads to better document labeling as well.

In the paper we followed[1], the author proposed an instance-based learning model called *Group-Instance Cost Function* (GICF). Basically, the algorithm was to perform a gradient descent logistic classification while applied with a brand new loss function which could simultaneously capture how much a sentence resembles the group semantically and evaluate similarity of sentences in the same group. The author performed evaluation experiment on the sentiment polarities predicting task, which is to classify a document as sentimentally positive or negative. It's a crucial topic in natural language processing for it is of great significance to

obtaining public opinion in social media nowadays[3]. There are twofold goals in the research where the first one is to propagate information from the review labeling to the sentences, while taking advantages of similarity of sentences within a review. The another one is to predict the label for groups by aggregating inferred sentence labels.

In our implementation, text formed data from Amazon and IMDb with 48,000 and 25,000 reviews respectively was used for training and testing. We performed doc2vec[4] as the sentence embedding strategy to convert preprocessed text into semantic related vectors at instance level. We trained the classifier using the code given by the author and after fine tuning hyperparameters, a test accuracy of 80.4% was achieved at instance level classification and 85.28% at group level classification on IMDb dataset, while 83.8% and 83.13% on Amazon dataset, respectively. Comparing to the reference results given by GICF paper[1], these results were kind of inferior with a disparity of about 3% to 10%. We attributed these differences to the different embedding model in the reproducing pipeline. It was also found that momentum has rather significant impact on model performance. Later an ablation study was carried out to evaluate effects of different parts in the cost function and we were surprised that removing the kernel function in the similarity cost didn't bring inferior results. Furthermore, different aggregation functions at group level prediction were also experimented.

2 Related Work

In this well-studied task of sentiment classification, different perspectives and approaches were used from basic machine learning techniques such as Naive Bayes and SVM[5] to recurrent neural networks[6]. Here we focused on researches that are relevant to instance based sentiment classifications (rather than group to group). In a traditional multiple instance supervised(MIL) learning problem, a group is defined as negative if all instances are negative and positive otherwise[2]. Although the method is widely used in particular problems, for instance, object localization in image, it is still necessary to relax the unbalanced restriction of MIL and assume all instances contribute independently to a group's label[7] in sentiment analysis. A number of other forms of aggregation were performed. Weidmann [8] consider a generalization where the presence of a combination of instance types determines the label of the group. Xu [9] assume that all instances contribute equally and independently to a group's class label. Zhou [10] build a model that solves MIL through semi-supervised learning techniques by considering a negative label for every instance in a negative group. Here, the followed paper author utilized averaging aggregation in prediction of a group.

Another point in this paper is the sentence embedding. Sentence embedding is the learning approaches to automatically convert each sentence to a vector for further research which can be completed through encoder and decoder models[11] or convolutional neural networks[12]. Doc2vec, i.e. *Paragraph Vector*[4] is an excellent tool for sentence embedding, which can better deal with text labeling with variable lengths and has got an error rate of only 12.2% on Stanford Sentiment Treebank Dataset.

3 Dataset

In GICF paper, the researchers did experiments of their model on three real world datasets from either Amazon, IMDb or Yelp, as well as their manually labeled instance dataset. Amazon dataset[13] contains cellphone reviews and scores(an integer scaling from 1 to 5), for which they considered a score of 4 or 5 as positive ,and 1 or 2 as negative. IMDb[14] contains 50000 movie reviews which were labeled either positive or negative. Yelp dataset provides restaurant reviews information for both image classification and sentiment analysis. The author extracted Yelp reviews in text format and scores with the identical scale and processing technique with Amazon dataset. Since each review(group) in the three datasets consists of several sentences(instances) and a binary label, they are suitable for the task of learning and predicting from groups to groups. But here the author want to investigate the relationship between instance labeling and group labeling. So for each dataset resources above, 1,000 new sentences instances of 500 positive and 500 negative were artificially labeled and thus to evaluate instance-level classification.

In our reproduction, we downloaded the same original Amazon and IMDb dataset as well as the author labeled instance dataset. Due to the overlarge size of Yelp dataset of over 600,000 reviews, we didn't introduce it into our work. Specifically, Amazon dataset has 48,000 groups in total containing of about 320,000 instances, which is, 6.76 instances per group on average. Meanwhile, IMDb consists of 25,000 groups and

more than 360,000 instances, with 14.72 instances per group on average. For each dataset, we partitioned it randomly into two parts, 80% as training set and 20% as test set. To be fairly evaluated, each dataset contains approximately equal proportion of positive groups and negative groups. We also imported the author labeled instances datasets corresponding to IMDb and Amazon. They were selected to evaluate the model performance on sentence level.

4 Data Preprocessing and Sentence Embedding

4.1 Data Preprocessing

All the datasets mentioned above were initially stored in individual files and contained some messy information which are irrelevant to sentiment classification. So data preprocessing is necessary. Before sentence embedding conversion, the datasets were first read into two python lists and stored in *.csv* files for faster data loading. Subsequently meaningless HTML tags and special characters were removed. Moreover all text were converted into lowercase, which are significant for implementing the following algorithms afterwards.

4.2 Sentence Embedding

In order to finish a group classification on basis of instance classification, it is necessary to convert sentence texts into learnable features. Actually, the embedding model has a strong affects to the final learning results. Therefore, we want to select sentence embedding techniques to represent each sentence(instance) as a fixed length vector which can capture the semantic relationships of different sentences. From this perspective, traditional embedding for natural language processing such as Bag of Words, n-gram, tf-idf are not suitable as they can not reflect semantic information.

In the original paper, the author used ConvNet for embedding task. ConvNet is a model of Denil et al.[15] which implemented a supervised multi-level convolutional neural network that can transform both sentences and documents into a vector in an embedding space. This model can accurately represent semantically similar sentences into close vectors in the embedding space. Since it can represent a sentence without sentence-level label, it is suited to the task setting. However, this method is only proposed in the additional paper [15] and lack of any implementation details. Due to the complicated implementation and high computational cost, we did not choose this model into our pipeline.

We investigated different alternative sentence embedding methods which use the geometry of a continuous embedding space and selected doc2vec as the most reasonable one. Doc2vec is an adaptation of word2vec which is to use the word2vec model and add another document vector. When training the word vectors, the document vector is trained as well, and in the end of training, it holds a numeric representation of the document. It is an unsupervised algorithm to generate distributed representations for sentences or documents. The vectors generated by doc2vec can be used for tasks like finding semantic similarity between sentences or documents. In this aspect, we think it has fairly high feasibility and due to its convenient implementation from Gensim[16], we selected doc2vec as our sentence embedding method. Each instance vector has a length of 100.

5 Proposed Method and Evaluation

With sentence vector features, the subsequent objective is to create and implement a new learning algorithm that can better reflect the relationship between groups and instances. Then certain classifier should be selected to predict the label of instance and aggregation method should be used for predicting group polarity.

5.1 Proposed Algorithm——GICF

Group instance cost function (GICF) is the main novel algotrithm proposed by the original paper[1]. According to the paper, it is vital to "propagate information from group labeling to instances", which gives an inspiration that an instance should have the same label with the group if their semantic structures are similar and, furthermore, close sentence vectors should lead to same instance labels.

Consequently the author suggested a new loss function with which we can easily learn a set of parameters θ using mini-batch stochastic gradient descent approach and predict instance labels. The cost function optimized is:

$$J(\theta) = \frac{1}{N^2}\sum_{i=1}^{n}\sum_{j=1}^{n} e^{-\|x_i - x_j\|_2^2}(\sigma(\theta^T x_i) - \sigma(\theta^T x_j))^2 + \frac{\lambda}{K}\sum_{k=1}^{K}(\frac{1}{|G_k|}(\sum_{i \in G_k}\sigma(\theta^T x_i)) - l_k)^2$$

where, x_i and x_j are instances from a group, G_k is a specific group with several instances, N and K are total numbers of instances and groups respectively, θ is set of parameters to extract information from the sentence vector, l_k is the true label of the kth group and λ is the balance parameter to adjust proper contribution of the two terms (similarity cost and group cost).

Both two terms of this loss function are essential to finish this instance classification task. As the author pointed out, the first term "spreads label information over the data manifold in feature-space"[1], which means to evaluate similarity of each pair of instances to complete such semi-supervised-like problem where we know labels but lack of instance labels. The term $e^{-\|x_i - x_j\|_2^2}$ is a special transformation of Euclidean norm and radial basis function(RBF) set up by the author, which can accurately represent similarity of two high-dimension vectors. Meanwhile the second term is critical to restrict instances semi-supervised-like learning with the group label and to avoid cases where all labels are predicted with the same label, ignoring the true label of the group.

With the aforementioned loss function, our task is to perform a mini-batch gradient descent learning process, try to optimize well fitting parameter θ and use it to make instance predictions.

5.2 Evaluation Methodology

After constantly tuning hyperparameters (learning rate, optimizer functions, batch size, etc.) and training on a large number of group-level data sets, we got the corresponding parametersθ. Since our final goal is to apply this model to both group level (reviews) and instance level (sentences within reviews), we need to make label prediction for these two parts.

5.2.1 Instance-level Prediction

For an individual sentence, we fed it into our doc2vec model and obtained a fixed length vector X. Then the instance-level prediction is rather intuitive and can be directly obtained by calculating logistic regression function.

$$\hat{y_i} = \hat{y_\theta}(x_i) = \sigma(\theta^T x_i) = \frac{1}{1 + e^{-\theta^T x_i}}$$

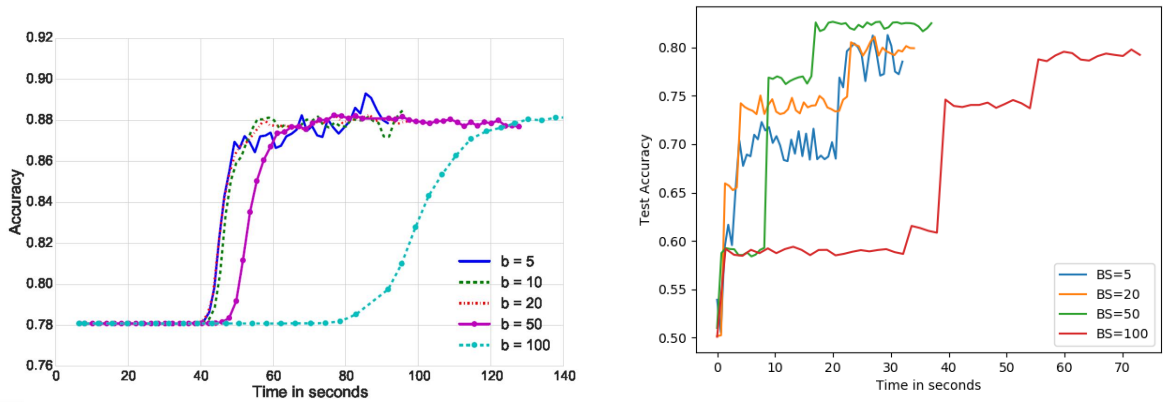
where $\hat{y_i}$ denotes the predicted label of a sentence x_i.

5.2.2 Group-level Prediction

Computing group-level label prediction is more complicated and indirect. According to algorithm mentioned above, it is impossible for us to directly compute group-level polarity without utilizing instance-level prediction. Thus a proper aggregation method should be selected to combine polarities of instances within the group. The GICF paper[1] used instance classification averaging as the group prediction. Besides this, we adopted two more aggregation methods for exploration. More details and results are in section 7.2.2.

6 Reproduced Results and Discussion

To have a comprehensive understanding towards the GICF model[1], we first give the best performances of the model we reproduced, compare it with the original results, then illustrate our hyperparameters tuning procedure as well as our understanding and evaluation of each parameter in the model.



(a) Paper presented, for the Yelp Dataset

(b) Our implementation, for the Amazon Dataset

Figure 1: Accuracy vs Time in Seconds, for different batch sizes

6.1 Performance Comparison

The author didn't provide hyperparameters choosing in the paper. After model testing and parameter tuning, we reached our best model using hyperparameters in Table 1. Details on functionality of each parameter will be discussed in section 6.2.

Batch Size	Similarity fn	Learning Rate	Momentum	Lambda Balance	Min instances/group
50	RBF	0.05	0.7	0.1	3

Table 1: Model Parameters

Table 2 and 3 shows the accuracy and AUC scores of our model at the group and instance level respectively. It is shown that our model achieved a relatively good performance, nonetheless our accuracy is lower than the one in the original paper by about 3% to 9% on two datasets. This is very likely due to different sentence embedding methods of our models. The embedding method of ConvNet used by the author was actually supervised, while we used unsupervised embedding method of doc2vec. Although both ConvNet and doc2vec provide a good function to map semantically similar sentences into nearby points in embedding space, the information they captured probably have different characteristics and therefore ConvNet's embedding results match better with GICF model.

Another notable issue is that accuracy at the instance level is slightly lower than group level accuracy which has similar performance with the original paper because the learning process is supervised on group level and the group predictions synthesize many instances' prediction.

We also reproduce the result of scalability validation, as shown in Figure 1. Though the test was based on different dataset as we didn't import Yelp dataset, we got the same conclusion that the algorithm was suitable to do mini-batch training. The small batch size choosing can significantly increase the training speed and achieve similar training effects. That shows the large range of scalability of the algorithm.

	Acc Train	Acc Test	Acc (Paper)	AUC Test	AUC (Paper)
IMDb	80.03%	85.28%	88.56%	84.40%	88.36%
Amazon	81.87%	83.15%	92.80%	83.08%	91.73%

Table 2: Accuracy and Area-Under-the-Curve (AUC) scores at the group (document) level

	Accuracy	Accuracy(Paper)	AUC	AUC(Paper)
IMDb	80.4%	86.0%	81.26%	75.94%
Amazon	83.8%	88.2%	83.82%	89.55%

Table 3: Accuracy and Area-Under-the-Curve (AUC) scores at the instance(sentence) level

6.2 Hyperparameters Reliability

6.2.1 Momentum

During our experiment, we firstly tested with the momentum coefficient in our model. The author defined a new parameter iterating momentum, and accordingly, the parameter updating rule is now:

$$w_{j+1} = \eta * w_j - \alpha \frac{\delta Err}{\delta w_j}$$

where w_j refers to the jth iteration of parameters. The formula intuitively indicates that a smaller momentumη means the cost gradient has a larger proportion of influence to the parameter updating whereas a larger η make the original parameter value hold a larger impact on parameter iterations. After testing with different values, as the result shown in Table 4, we found that an η of 0.7 gives the best performance on both Amazon and IMDb dataset. With $\eta = 1$, this is a pure gradient descent learning process and performs rather bad on Amazon dataset.

Momentum	0.3	0.7	0.9	1
IMDb	80.36%	85.28%	84.06%	84.78%
Amazon	82.40%	83.15%	64.30%	60.47%

Table 4: Group level accuracy with different momentum

6.2.2 Minimum Instance

Minimum instance is an important field to filter out groups with few number of instances which could not reflect the functionality of multi-instance learning. Although this is not part of model enhancing, it can still provide us with an overview of the model performance, as shown in Table 5. As aforementioned, IMDb has 14.72 instances per group on average and thus there is no obvious fluctuation in accuracy as the minimum instance changes, which proves this multi-instance classification has successfully dealed with aggregation of many instances in a group. However, Amazon dataset has only 6.76 instances on average and there are insufficient groups left after removal of groups with less than 8 instances. That may cause the decreasing of accuracy. Also, when the model were performed on many short groups it cannot take advantages of instance similarity and the aggregation, so minimum instances of 1 also brought the inferior result.

Minimum Instances	1	3	8
IMDb	84.78%	85.28%	86.00%
Amazon	78.45%	83.15%	68.77%

Table 5: Group level accuracy with different minimum instances per group

6.2.3 Lambda Balance

As stated in section 5.1, there is a balance parameter λ in the group cost term to configure the proportion of the two terms. With result of this experiment shown in Table 6, it confirms that with $\lambda = 0.1$, both datasets achieve a relatively high accuracy. However, modifying this parameter does not show a significant improvement in model performance.

Lambda Balance	0.01	0.05	0.1	0.5	2.5
IMDb	83.28%	83.64%	85.28%	84.54%	83.68%
Amazon	79.68%	79.46%	83.15%	83.49%	81.73%

Table 6: Group level accuracy with different lambda balance

7 Further Exploration

7.1 Cost Function Ablation

In this section, we performed a detailed ablation on different parts of the cost function to find which part has more significant contribution. As mentioned in section 5.1, since the cost function includes two terms in total (group cost and similarity cost), we considered two ablation steps: group cost only, similarity cost only. The results are shown in Table 7. From the result we found that neither of these ablation experiments beats the original GICF cost function, which means that both group cost and similarity cost are essential to our final result. However, it is obvious that the group cost is more important relative to the similarity cost.

We also experimented different similarity kernel functions in the cost function: RBF and cosine similarity(COS). COS measures the cosine of the angle between two non-zero vectors to represent their similarity. The results compared with the algorithm without kernel function are shown in Table 8. This comparison demonstrates that the similarity cost can adequately satisfy our goal without kernel function.

	IMDb	Amazon
Group Cost Only	83.68%	77.41%
Similarity Cost Only	73.36%	64.18%
GICF (Paper Proposed)	85.28%	83.15%

Table 7: Cost Function Ablation Test

Similarity Function	IMDb	Amazon
RBF (Paper Proposed)	85.28%	83.15%
COS	82.64%	80.42%
Without Kernel	84.78%	83.18%

Table 8: Similarity Function

7.2 Group Aggregation Exploration

As mentioned in section 5.2.2, to obtain group label prediction, we tried several different aggregation methods to combine polarity information of sentences within a group. The first method is to average the classification possibilities given by logistic regression function, which is exactly the same as mentioned in the original paper. While for the second one, we labeled each instance, for example, if the output of logistic regression function is more than 0.5, then labeled it as positive, otherwise negative. Subsequently we performed majority vote among all the instances within the group to obtain the final group-level prediction. And for the last one, we directly took the mean of minimum and maximum of the instances. The performance of evaluation with these three aggregation methods are shown in Table 9, which indicates that the averaging algorithm proposed by the original paper is reasonable and proper.

	IMDb	Amazon
Averaging (Paper Proposed)	85.28%	83.15%
Median (Voting)	82.32%	79.76%
Mean of min and max	80.68%	81.92%

Table 9: group evaluation

8 Conclusion

In this task, we reproduced the algorithm introduced by GICF paper[1] using the same dataset and different embedding pipeline. After fine tuning relevant training hyperparameters of this GICF model, we basically obtained the optimal parameters and achieved the goal of this task, which is to predict labels for both groups(reviews) and instances(sentences) when given labels of groups. We gained a final accuracy of 85.28% at group level and 80.4% at instance level on IMDb dataset, meanwhile 83.15% at group level and 83.8% at instance level on Amazon dataset. The performance of the our model achieved fairly high accuracy and a close distribution with the original performance.

To further explore the reliability and robustness in this model, we performed a detailed ablation on both group and similarity parts of the cost function and found that the group cost contributes more to the whole model. However, the kernel function is replaceable. Finally, we also implemented an exploration on group prediction. We tried different methods to aggregate instances labels into a group label, and thus it is illustrated that the averaging algorithm proposed by the original paper is proper.

9 Statement of Contribution

Andi Dai implemented and tuned the GICF code and was responsible for reproduced results, discussion and further exploration part as well as table preparation in the report and proposed approach section of this report. Yiran Mao implemented data preprocessing and sentence embeddings, wrote dataset, preprocessing, sentence embedding and proposed method sections of this report. Jingyuan Wang tested with ablation study and further improvement and wrote other sections in the writeup.

References

[1] Dimitrios Kotzias, Misha Denil, Nando de Freitas, and Padhraic Smyth. From group to individual labels using deep features. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 597–606, New York, NY, USA, 2015. ACM.

[2] Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71, January 1997.

[3] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135, January 2008.

[4] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014.

[5] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural*

Language Processing - Volume 10, EMNLP '02, pages 79–86, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[6] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Recurrent neural network for text classification with multi-task learning. *CoRR*, abs/1605.05101, 2016.

[7] Xin Xu and Eibe Frank. Logistic regression and boosting for labeled bags of instances. In Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang, editors, *Advances in Knowledge Discovery and Data Mining*, pages 272–281, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[8] Nils Weidmann, Eibe Frank, and Bernhard Pfahringer. A two-level learning method for generalized multi-instance problems. In *European Conference on Machine Learning*, pages 468–479. Springer, 2003.

[9] Xin Xu and Eibe Frank. Logistic regression and boosting for labeled bags of instances. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 272–281. Springer, 2004.

[10] Zhi-Hua Zhou and Jun-Ming Xu. On the relation between multi-instance learning and semi-supervised learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1167–1174. ACM, 2007.

[11] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[12] Misha Denil, Alban Demiraj, and Nando de Freitas. Extraction of salient sentences from labelled documents. *CoRR*, abs/1412.6815, 2014.

[13] Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 165–172, New York, NY, USA, 2013. ACM.

[14] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 142–150, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[15] Misha Denil, Alban Demiraj, and Nando de Freitas. Extraction of salient sentences from labelled documents. *CoRR*, abs/1412.6815, 2014.

[16] Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.