`uniblock`: Scoring and Filtering Corpus with Unicode Block Information

Yingbo Gao **Weiyue Wang** **Hermann Ney**
Human Language Technology and Pattern Recognition Group
Computer Science Department
RWTH Aachen University
D-52056 Aachen, Germany
`<surname>@i6.informatik.rwth-aachen.de`

Abstract

The preprocessing pipelines in Natural Language Processing usually involve a step of removing sentences consisted of illegal characters. The definition of illegal characters and the specific removal strategy depend on the task, language, domain, etc, which often lead to tiresome and repetitive scripting of rules. In this paper, we introduce a simple statistical method, `uniblock`[1], to overcome this problem. For each sentence, `uniblock` generates a fixed-size feature vector using Unicode block information of the characters. A Gaussian mixture model is then estimated on some clean corpus using variational inference. The learned model can then be used to score sentences and filter corpus. We present experimental results on Sentiment Analysis, Language Modeling and Machine Translation, and show the simplicity and effectiveness of our method.

1 Introduction

Identification and removal of sentences with illegal characters is a common heuristic in the preprocessing pipelines in Natural Language Processing (NLP). While it has benefits of controlling the vocabulary size and dropping noisy data, it is often a tedious work to come up with appropriate rules and removal strategies when facing with different tasks, languages and domains. The lack of clear definition of what is illegal exacerbates the problem. For example, modern Chinese text may allow characters such as: traditional and simplified Chinese characters, special punctuation marks, full-width characters, emojis, mathematical symbols, Latin characters, currency symbols, scientific notations, etc. As a result, scripting robust rules often requires a considerable amount of time and effort.

In this paper, we introduce a simple statistical method, `uniblock`, to address the problem. The motivation of our approach is straightforward - good rules may vary greatly depending on the situation, therefore instead of designing rules by hand, we use "rules" defined by data. We assume that some clean corpus is available, in which all the characters are deemed legal and the character distributions are similar to the test cases. It is possible to learn a probabilistic model, which describes the clean corpus and assigns scores to sentences in another corpus. The scores can further be used for filtering. Since the legalness of characters is in question, Unicode block information is a natural choice for obtaining feature vectors. Note that, by designing alternative feature vectors, one can potentially adapt `uniblock` to implement other corpus filtering heuristics.

We develop `uniblock` mainly for Machine Translation (MT). It can also be easily applied to other NLP tasks. We present experimental results on Sentiment Analysis (SA), Language Modeling (LM) and MT, and show the simplicity and effectiveness of our method.

2 Related Work

Raw data in NLP is often noisy. Khayrallah and Koehn (2018) categorize five common noise sources in parallel corpora and count only about 23% of the sentences in the raw 2016 ParaCrawl corpus[2] to be "Okay". Although illegal characters is not listed as a separate noise source, misuse of characters and shifting of character distributions may result in a sentence being classified into one of the five noise sources. In previous work, a supervised model using bag-of-words translation features is developed to classify clean and noisy data (Xu and Koehn, 2017; Khayrallah et al.,

[1] The source code is available at `https://github.com/ringoreality/uniblock`

[2] `https://paracrawl.eu/index.html`

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 1324–1329,
Hong Kong, China, November 3–7, 2019.

2018). In contrast, our model, which is trained in an unsupervised manner, tackles the illegal character problem explicitly.

Koehn et al. (2018) describe a shared task on parallel corpus filtering. While participating systems focus on addressing both monolingual fluency and bilingual adequacy, character-level filtering is common to all submissions. Junczys-Dowmunt (2018) applies a language identification model to implicitly remove sentences with illegal characters. Rossenbach et al. (2018) keep sentences with more than three words, with each word having at least one character from the predefined alphabet of the language. Lu et al. (2018) remove characters outside of a predefined alphabet. Ash et al. (2018) count most frequent characters, set a cutoff around eighty for each language, and remove sentences with illegal characters. Erdmann and Gwinnup (2018) get rid of lines containing characters from the Unicode general category of "other". Papavassiliou et al. (2018) simply consider Latin Unicode characters to be legal.

Unicode is the de facto standard for encoding characters from various languages, domains and sources (The Unicode Consortium, 2019). It uses "blocks" to group characters with similar origins or functions. The current version 12.0 defines 300 blocks, including Basic Latin, Latin-1 Supplement, CJK (Chinese, Japanese and Korean) Symbols and Punctuation, etc. To identify the legalness of characters, the Unicode block information provides meaningful discriminative signals.

The Gaussian Mixture Model (GMM) is a classic algorithm that assumes data is generated from a mixture of finite number of Gaussian distributions, whose parameters are typcially estimated with the Expectation–Maximization (EM) algorithm. An extension to the EM algorithm is variational inference, which has the advantage of automatically choosing the number of components. Bishop (2006) gives a comprehensive introduction to the topic. We use the implementation of variational Bayesian estimation of Gaussian mixtures from `scikit-learn` (Pedregosa et al., 2011).

3 Methodology

3.1 Feature Vectors

In order to assign meaningful scores to sentences and eventually filter out those who contain illegal characters, we first need to design appropriate features. We believe, the Unicode block information has a good property that the blocks are grouped by origin and function. For instance, CJK Symbols and Punctuation has a dedicated Unicode block in range U+3000...U+303F. Specifically, we count the appearances of characters in each of the 300 blocks. This will result in feature vectors $c_1, c_2, ..., c_B$ with a fixed length of $B = 300$. If we further normalize them by the total number of characters in each sentence, probability distributions over the 300 blocks can be obtained. We observe that there are only few blocks whose counts are non-zero in natural language texts. This calls for dimensionality reduction methods. Empirically, we drop the zero-count dimensions directly during training and assign conceptually low scores[3] when a non-zero count is seen during scoring. That is, we use normalized feature vectors $e = e_1, e_2, ..., e_{B'}$, where $1, 2, ..., B'$ are dimensions in B whose original counts are non-zero, for training.

3.2 Bayesian Gaussian Mixture Model

Although many corpora are noisy, it is not appropriate to deem all sentences in them "dirty". While generating synthetic noisy data is always an option, it is unrealistic to cover all types of noises. Compared to the difficulty to obtain negatively labelled data, the development set is often available and can be deemed "clean" with high confidence. Therefore, we take the development set as training data for our scoring system and treat the problem as a clustering task rather than a classification task.

We assume that the training feature vectors e are generated by a mixture of Gaussian distributions.

$$\begin{aligned} p(e) &= \sum_{k=1}^{K} \pi_k \mathcal{N}(e|\mu_k, \Sigma_k) \\ \pi_k &\sim \mathcal{DP}(\alpha) \\ \mu_k &\sim \mathcal{N}(\mu_0, \sqrt{\frac{1}{\tau}}) \\ \Sigma_k &\sim \mathcal{W}_{B'}(V, n) \end{aligned} \tag{1}$$

In the equation above, k is a running index in the number of mixtures K, π_k is the mixture weight for the k-th Gaussian, μ_k is a B'-dimensional vector parametrizing the k-th mean, Σ_k is the k-th $B' \times B'$ covariance matrix. We further impose priors on the model parameters: π_k follows a Dirichlet Process ($\mathcal{DP}$), which is parametrized by concetration prior α; μ_k follows a Normal distribution

[3] zeros in our experiments

($\mathcal{N}$), which is parametrized by the B'-dimensional mean prior μ_0 and the precision prior τ; Σ_k follows a Wishart distribution ($\mathcal{W}$) in B', which is parametrized by the covariance prior V and degree of freedom prior n. We estimate the model parameters using the EM algorithm. Note that operating in B' dimensions leads to significantly better model convergence than training in B dimensions.

3.3 Scoring and Filtering

Once the model is trained till convergence, it is possible to use it to assign scores to unseen feature vectors. We directly use the weighted log probabilities as scores. Compared to the sentences used during `uniblock` training, higher scored sentences have more similar Unicode block count distributions, or fewer illegal characters. Regarding how to remove noisy sentences, we implement two straightforward ideas: absolute thresholding and relative thresholding. The former removes sentences with scores lower than a given threshold while the later removes a given percentage of low-scored sentences. We also add support for filtering parallel data. A simple method is to combine scores across parallel sentences with a specified reduction method (minimum, maximum, average), and then apply thresholds. Alternatively, one can use the lowest score seen during training as an absolute threshold for each language and filter parallel data in a "one bad, remove all" manner.

4 Experiments

We conduct experiments on three NLP tasks: SA, LM and MT. As shown in related work, many different heuristics are used for the purpose of illegal character filtering, and there is no generally accepted "golden rule". Therefore, we focus on comparing "no filtering" versus "filtering", to examine how our method performs.

4.1 Sentiment Analysis

We conduct SA experiments on the Stanford Twitter Sentiment (STS) corpus (Go et al., 2009) using fastText (Joulin et al., 2017). Given 1.6M automatically annotated tweets, the task is to predict postive or negative sentiments on 498 test tweets. For the baseline, we use minimal preprocessing to handle casing, weblinks, usernames, hashtags and spacing in the raw training tweets, without explicit character-level filtering. For `uniblock`, we additionally train a GMM on the tweets in the test set[4] and use the minimum score as an absolute threshold to filter the training corpus. In total, about 0.9% training tweets are filtered out. We observe that only one Unicode block exists in the test set, which means all sentences with characters in other blocks are assigned conceptually low scores and removed. In this particular case, our general method reduces to a simple filtering rule similar to that of Papavassiliou et al. (2018). As shown in Table 1, our method improves the test accuracy over the baseline by 0.6%.

	test accuracy [%]
baseline	81.6
`uniblock`	**82.2**

Table 1: Test accuracies on STS.

4.2 Language Modeling

In MT pipelines, language models are often used for corpus filtering, fusion techniques and N-best reranking. Therefore, we perform LM experiments on MT corpora. Specifically, we take monolingual data from parallel corpora of WMT19 (Barrault et al., 2019) Chinese-English (zh-en) and Russian-English (ru-en) as training data. We use newsdev2017 for both sides of zh-en and newstest2016 for both sides of ru-en as development data. We concatenate newstest2017 and newstest2018 as test data for all four sides. The sizes of the corpora (WMT) are summarized in Table 2.

	zh-en	ru-en
train	26M	25M
valid	2K	3K
test	6K	6K

Table 2: Number of sentence pairs in WMT.

	zh-en		ru-en	
data	zh	en	ru	en
100%	40.9	135.0	181.0	**145.7**
90%	**36.9**	133.5	**176.7**	147.6
80%	37.1	132.9	180.6	149.3
70%	38.7	**131.4**	178.6	151.1

Table 3: Test perplexities on WMT.

[4] No development set is available for STS, this is why we trained the GMM on the test set. For LM and MT experiments, `uniblock` is trained on the development sets.

side	score	parallel text
zh	-48523.31	リリスノト
en	53.82	Open Directory - Unix
zh	67.71	健康、整洁、卫生和清洁技术组织
en	-158.57	Salubrité, propreté, hygiène et techniques d'assainissement
zh	0.00	从系统目录→位置→传送我的位置...
en	0.00	From System Menu → Location → Send My Location...
zh	36.01	在25℃下测定了样品的总交换容量和平衡等温线.
en	40.65	Cation exchange capacity and equilibria isotherm at 25 ℃ were determined.
zh	68.63	财务政策及发展小组
en	53.82	Financial Policy and Development Unit

Table 4: Several text samples from the zh-en parallel corpus with `uniblock` scores. We find empirically: sentences with only foreign characters are scored very low; sentences with some foreign characters are scored low; sentences with characters in unseen blocks are scored 0; sentences with only few foreign characters are scored high; sentences with no illegal characters are scored very high. Foreign (or illegal) characters are underlined.

For each of the four LM subtasks, we train `uniblock` on the corresponding development set and filter out 10%, 20% and 30% of the raw training corpus. We train 2-layer LSTM language models with hidden dimensions being 1024. For the zh subtask, we train character-level language models and include all seen characters in the vocabulary. For the other 3 subtasks, top 100k frequent words are used as the vocabulary and the rest of the words are replaced with unknown tokens. All LM experiments are conducted using the RETURNN framework (Doetsch et al., 2017; Zeyer et al., 2018). As shown in Table 3, we see consistent improvements in the `uniblock`-filtered setups of zh, ru and en under zh-en. For en under ru-en, the baseline system outperforms `uniblock`-filtered ones.

4.3 Machine Translation

We train MT systems to show the simplicity and effectiveness of our method in a more challenging task. We use the same parallel corpora as in Section 4.2 and train systems in four directions: zh-en, en-zh, ru-en and en-ru. The raw training corpora are filtered with `uniblock` as before. Before examining the performance of the translation systems, we first manually examine some filtered text samples. Five parallel text samples with `uniblock` scores are shown in Table 4. We observe that the trained GMM works properly and indeed assigns low scores to unreasonable or "dirty" sentences. We believe that `uniblock` is superior to other rule-based methods because the relative quality of the sentences is modeled. For example, a hard rule may eliminate a complete sentence pair encountering even one ℃ symbol, while `uniblock` only gives a small penalization due to the fact that one occurrence of the symbol ℃ does not alter the count-based feature vector e much.

	zh-en		en-zh	
data	test17	test18	test17	test18
100%	25.0	24.5	30.1	33.0
90%	**25.2**	**25.6**	**30.9**	33.1
80%	24.3	25.3	30.3	**33.2**
70%	24.3	24.8	30.2	33.0

	ru-en		en-ru	
data	test17	test18	test17	test18
100%	32.9	28.3	26.9	23.3
90%	**33.5**	**29.3**	**27.8**	**23.9**
80%	33.3	28.9	27.2	23.6
70%	32.9	28.1	26.6	23.2

Table 5: Case-sensitive BLEU(%) scores on four different translation tasks.

For all MT systems, we perform minimum preprocessing to handle the whitespaces and casing, and use SentencePiece (Kudo and Richardson, 2018) to obtain the subword units. All models adopt the Transformer (Vaswani et al., 2017) architecture and use the exact same hyperparameters as the base model. Trainings are done with the Sockeye (Hieber et al., 2017) toolkit and share the same optimization hyperparameters. No synthetic data is used and no ensembling is done during decoding. The only difference across the models is the training corpus - the baseline model uses the full parallel corpus, while the ones filtered by uniblock use a subset of the full corpus. We take the checkpoint with the lowest perplexity on the validation set and report case-sensitive BLEU(%) (Papineni et al., 2002) scores on newstest2017 and newstest2018 using the sacreBLEU (Post, 2018) software. The translation qualities of the systems are shown in Table 5. As can be seen, without altering the architecture or the optimization process, only applying uniblock as a corpus filtering step leads to consistent improvements over the baseline systems for all four directions, up to 1.1 BLEU(%).

Linguistically, the zh-en and ru-en language pairs are rather distant. For more closely-related language pairs such as French and English, our method should also perform well. As shown in Table 4, a Chinese sentence could be corrupted with Japanese characters and an English sentence could be corrupted with French characters. For both cases, our method is able to discriminate the illegal characters and assign low scores to these sentences. Note that, our method would fail, if "clean" and "dirty" sentences share one exact same block. Because then after normalization all the feature vectors will be essentially the same and therefore indistinguishable. In this case, one should carefully design other features so that the GMM is able to assign meaningful scores. However, when "clean" and "dirty" sentences share the same set of blocks, our method still works fine because after normalization the empirical distributions are probably different.

5 Conclusion

In this work, we develop a simple and effective statistical method called uniblock. Using Unicode block information as feature vectors, a GMM is estimated with variational inference on some clean corpus, which can then be used to score and filter corpus. We release our implementation which supports parallel corpus filtering and different thresholding. Our experiments show concrete and consistent improvements in SA, LM and MT.

We believe that the method can be extended and improved by examining other dimensionality reduction methods and alternatives to GMM, and by introducing other heuristics into the feature vector, such as sentence length and punctuation marks[5].

[5] Common punctuation marks lie in the same Unicode block range U+0000..U+007F as the English alphabet and are currently not separated.

Acknowledgements

This work has received funding from the European Research Council (ERC) (under the European Union's Horizon 2020 research and innovation programme, grant agreement No 694537, project "SEQCLAS") and the Deutsche Forschungsgemeinschaft (DFG; grant agreement NE 572/8-1, project "CoreTec"). The GPU computing cluster was supported by DFG (Deutsche Forschungsgemeinschaft) under grant INST 222/1168-1 FUGG.

References

Tom Ash, Remi Francis, and Will Williams. 2018. The speechmatics parallel corpus filtering system for wmt18. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 853–859.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (wmt19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. Association for Computational Linguistics.

Christopher M Bishop. 2006. *Pattern recognition and machine learning*. springer.

Patrick Doetsch, Albert Zeyer, Paul Voigtlaender, Ilia Kulikov, Ralf Schlüter, and Hermann Ney. 2017. Returnn: The rwth extensible training framework for universal recurrent neural networks. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5345–5349. IEEE.

Grant Erdmann and Jeremy Gwinnup. 2018. Coverage and cynicism: The afrl submission to the wmt 2018 parallel corpus filtering task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 872–876.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12):2009.

Felix Hieber, Tobias Domhan, Michael Denkowski, David Vilar, Artem Sokolov, Ann Clifton, and Matt Post. 2017. Sockeye: A toolkit for neural machine translation. *arXiv preprint arXiv:1712.05690*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

Marcin Junczys-Dowmunt. 2018. Dual conditional cross-entropy filtering of noisy parallel corpora. *arXiv preprint arXiv:1809.00197*.

Huda Khayrallah and Philipp Koehn. 2018. On the impact of various types of noise on neural machine translation. *arXiv preprint arXiv:1805.12282*.

Huda Khayrallah, Hainan Xu, and Philipp Koehn. 2018. The jhu parallel corpus filtering systems for wmt 2018. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 896–899.

Philipp Koehn, Huda Khayrallah, Kenneth Heafield, and Mikel L Forcada. 2018. Findings of the wmt 2018 shared task on parallel corpus filtering. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 726–739.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Jun Lu, Xiaoyu Lv, Yangbin Shi, and Boxing Chen. 2018. Alibaba submission to the wmt18 parallel corpus filtering task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 917–922.

Vassilis Papavassiliou, Sokratis Sofianopoulos, Prokopis Prokopidis, and Stelios Piperidis. 2018. The ilsp/arc submission to the wmt 2018 parallel corpus filtering shared task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 928–933.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

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.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Nick Rossenbach, Jan Rosendahl, Yunsu Kim, Miguel Graça, Aman Gokrani, and Hermann Ney. 2018. The rwth aachen university filtering system for the wmt 2018 parallel corpus filtering task. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 946–954.

The Unicode Consortium. 2019. The unicode standard, version 12.0.0. `http://www.unicode.org/versions/Unicode12.0.0/`.

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 *Advances in neural information processing systems*, pages 5998–6008.

Hainan Xu and Philipp Koehn. 2017. Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2945–2950.

Albert Zeyer, Tamer Alkhouli, and Hermann Ney. 2018. Returnn as a generic flexible neural toolkit with application to translation and speech recognition. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 128–133, Melbourne, Australia.