Query Language Identification with Weak Supervision and Noisy Label Pruning

Sweta Sharma
sharswet@amazon.com
Amazon Inc
Bengaluru, Karnataka, India

Vijay Huddar
vhhuddar@amazon.com
Amazon Inc
Bengaluru, Karnataka, India

Ishita Aggarwal
iaaggarw@amazon.com
Amazon Inc
Bengaluru, Karnataka, India

Namrata Khoriya
khorin@amazon.com
Amazon Inc
Bengaluru, Karnataka, India

Vishnu Narayanan
vishnun@amazon.com
Amazon Inc
Seattle, WA, USA

Atul Saroop
asaroop@amazon.com
Amazon Inc
Bengaluru, Karnataka, India

Rahul Bhagat
rbhagat@amazon.com
Amazon Inc
Seattle, WA, USA

ABSTRACT

Query Language identification is an important part of a multilingual product search system. However, accurate language identification in product searches is difficult due to multiple reasons, including presence of noise in available datasets. In this work, we propose a learning framework that combines weak supervision with noisy label pruning. We use Convolutional Neural Networks (CNN) based models to carry out such a combination. Our results show improvements over FastText baselines and FastText with weak supervision, thereby demonstrating the benefit of such a combination.

KEYWORDS

Weak Supervision, Language Identification, Ecommerce Search

ACM Reference Format:
Sweta Sharma, Vijay Huddar, Ishita Aggarwal, Namrata Khoriya, Vishnu Narayanan, Atul Saroop, and Rahul Bhagat. 2021. Query Language Identification with Weak Supervision and Noisy Label Pruning. In *Proceedings of Ljubljana '21: The Web Conference (Ljubljana '21).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

A typical multilingual e-commerce product search system is composed of the following sub-systems (as shown in Figure 1): 1) an inline search suggestion sub-system that guides customers in their search queries as they type them out in a search bar, 2) a speller that corrects any unintentional spelling mistakes, 3) a matcher - lexical, behavioral, or semantic - that generates a list of product matches for a customer query, and 4) a ranker that understands the latent intent of the query and ranks the output results accordingly. In a scalable product search system, each of these sub-systems are further divided into components to dissociate concerns, which can be built in parallel rather than in a monolithic manner. However, to function effectively (in terms of precision of search results generated) and efficiently (in terms of the number of resources used), each of these sub-systems needs to be aware of the language of the search query.

One simple approach to make all the sub-systems language-aware is to simply ask the customer to explicitly specify the query language. This approach, however, is only partially effective. Bilingually or multilingually literate customers, who may even be fluent in the primary language of the product search system, may find it difficult to recall the accurate and representative term to use for a product they are looking for. Less fluent speakers of the primary language may prefer to express their needs using code-mixed and transliterated queries. Less proficient customers may not even be able to accurately follow the process of setting the appropriate language of search using the user interface. Due to these reasons, evidence of a significant portion of code-mixed, transliterated and non-primary language content is found in samples of customer query traffic. Thus, query language identification is an important part of a multilingual product search system.

However, the problem of accurate language identification is difficult due to the following reasons: 1) absence of large high-quality datasets for language identification in product searches, 2) product searches being inherently short, 3) presence of long-tail in product searches in terms of their frequency of occurrence, and 4) presence of significant inherent and random noise in available datasets. In addition, languages originating from the same root language or forming a language group tend to share words, like in the case of Spanish, Portuguese and Italian.

In this work, we gainfully combine methods proposed in weak supervision and noise-handling sub-areas of machine learning for query language identification. We use weak supervision to obtain

Ljubljana '21, June 03–05, 2021, Ljubljana, Slovenia
© 2021 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

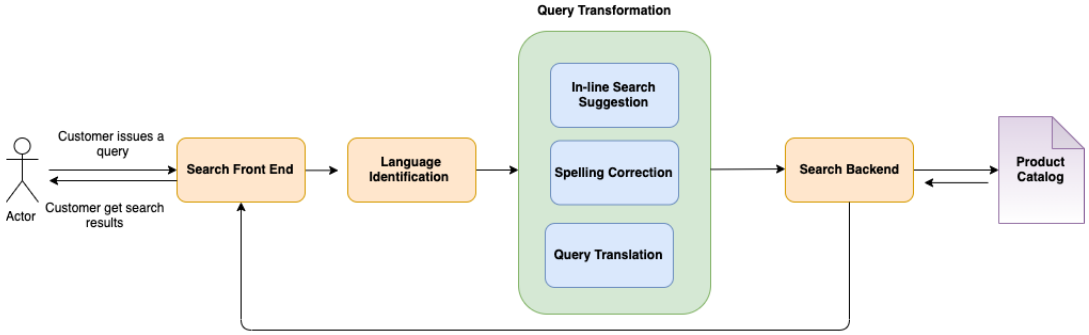


Figure 1: Flow chart describing the flow of a search query in a general search system

training data with noisy labels and use a CNN based noise-handling model to perform query language identification. To the best of our knowledge, this is the first reported work that combines weak supervision with noise handling for a multi-class classification for short text queries and is the primary contribution of our work. Our approach also contributes to the literature by achieving state-of-the-art results for query language identification in product searches.

The remainder of this paper is organized as follows: Section 2 gives a brief background of related work. In Section 3, we discuss our methodology that combines weak supervision and noise-handling for query language identification. Subsequently, Section 4 provides experiments and comparison with related methods. Section 5 has analysis and discussion on key observations. Finally, Section 6 gives concluding remarks.

2 RELATED WORK

Query language identification can be formulated as a standard text classification problem. Text classification is a well-researched area of natural language processing. Minaee et al. [8] explore more than 150 Deep Learning models for text classification. Some of these models achieve state-of-the-art results in various text classification tasks - sentiment analysis, news categorization, topic analysis, question answering, and natural language inference. However, Jauhiainen et al. [6] point out challenges and opportunities that make language identification different from generic text classification. For instance, as opposed to generic text classification, language detection algorithms need to pre-determine tokenization strategy prior to detecting a document's language. In addition, while in generic text classification, the class ratios are somewhat similar, language identification needs to operate on highly imbalanced datasets. As an example, high-quality sentences written in English language can be sourced easily from sports, fashion, politics, social media and e-commerce sites. To further exacerbate the problem, product queries tend to be shorter than generic text, and do not have widely available standardized golden datasets available for learning query language identification models.

Another area of work that is close to our problem is that of short text classification, which has gained popularity due to its applicability to social media like Twitter. Short text classification deals with a similar set of challenges as ours, i.e., short social media text typically lacks grammatical structure, is sparse, ambiguous, noisy, and riddled with spelling mistakes. Earlier work on short text classification used Naive Bayes and Support Vector Machines [11], while a majority of recent work focuses on neural models [16]. Wang et al. [16], articulate why explicit (part-of-speech tagging and knowledge bases) or implicit (neural representation) text representations used in standard text classification cannot effectively work for short text classification. Inspired by advances in usage of attention mechanism, Chen et al. [3] propose to include information from knowledge bases using knowledge-powered attention. However, a majority of approaches for classifying short text focus on methodology, while less emphasis is given to handling noisy and sparse labeled data.

Language identification in itself is a well-researched area. Multiple fast and efficient pre-trained language identification models and their associated implementations have been developed. FastText [7], Google's langdetect library [10], and Spacy[1] have shown promising results on generic language identification tasks based on open datasets. While Langdetect and Spacy use multinomial Naive Bayes, FastText uses neural models to compute sentence vectors by averaging n-gram embeddings and multinomial logistic regression for final language classification. Toftrup et al. [14] reproduced language identification architecture using bi-LSTMs, developed by Apple[2] and reported performance gains over existing open-source language identification algorithms. In their most recent work, Tambi et al. [13] perform search query language identification on the Adobe stock search system. In this work, they explore large-scale training data generation using weak supervision. They start with constructing a dictionary of words, MUSE dictionaries [4], and generate soft labels basis overlap between dictionary words and queries. To label unlabeled queries, they train a nearest neighbor algorithm using queries in the previous step.

In this paper, we explore weak supervision to generate labels using multiple algorithms and heuristics, and then refine our labels using a noisy pruning framework. Inspired by Xiao et al. [17] and Song et al. [12], we employ ideas based on learning with noisy labels. Also, we are aware of multiple approaches for merging labels from weak heuristic functions (Snuba [15], Snorkel [9], and Snorkel Drybell [1]). However, in this work we show that a combination of even

[1] https://spacy.io/universe/project/spacy-langdetect
[2] https://machinelearning.apple.com/research/language-identification-from-very-short-strings

a simplistic approach to weak supervision combined with noisy label pruning can lead to improved query language identification performance.

3 METHODOLOGY

In a typical search system, there is no lack of unlabeled data but acquiring corresponding label information is expensive, difficult and often prone to errors, as noted in Section 1. In view of aforementioned challenges along with this one, we propose an end-to-end methodology to train a multi-class language identification model with Noisy and Weak Labels (LINWL).

Starting with a set of n unlabeled queries, our proposed framework involves three phases. First, we combine the outputs from different labelling heuristics to obtain a single set of weak language labels for each query. We also obtain noise-type labels for each query-label pair to determine how likely a query has been labeled incorrectly. Next, we introduce a deep learning framework to identify labeled queries that are most likely to have label-noise and the associated type of noise. Finally, we use all queries that are identified to have clean labels (i.e. without label noise) for training of a query language identification model. Figure 2 provides a graphical representation of proposed framework for a multi-class query language identification model.

3.1 Obtaining Weak and Noisy Labels

Given a set of m labelling functions, $h_1, h_2, \ldots, h_m$ that generate (possibly noisy) language labels for queries, we produce a matrix of labelling function outputs, given by $\Lambda \in (\gamma \cup \{\phi\})^{n \times m}$ where γ represents the possible class ($\gamma = \{1, 2, \ldots, p\}$ for a p-class language identification problem; and ϕ denotes the case when a labelling function abstains from the labelling task. For simplicity, let the output of k-th labelling function be denoted by $l_{h_k}(x)$ i.e. $l_{h_k}(x) \in \{1, 2, \ldots, p\}\}$.

Our next goal is to convert the label matrix containing overlapping and conflicting labels for each query into a single vector of probabilistic labels $Y = (\hat{y}_1, \ldots, \hat{y}_m)$, where $\tilde{y}_i \in \{1, \ldots, p\}$ for a p-class classification problem. To do so, we use the following methodology to assign a weak label $\hat{y}$ for a query x:

$$\hat{y} = l_w(x) = \begin{cases} mode(\{l_{h_k}(x) : \forall h_k(x) \text{where } l_{h_k}(x) \in \gamma\}), \\ \qquad \text{if there are no ties} \\ \{l_{h_k}(x) : h_k \text{ has best accuracy amongst labelling} \\ \qquad \text{functions with 100\% labelling coverage }\}, \\ \qquad \text{in case of ties} \end{cases} \tag{1}$$

Let the resulting weakly labeled dataset be represented as $D_n = \{(x^{(1)}, \hat{y}^{(1)}), \ldots, (x^{(N)}, \hat{y}^{(N)})\}$ with N-th query represented as $x^{(N)}$ and its corresponding noisy label as $\hat{y}^{(N)}$. Here, we assume that the labels obtained by combining different heuristics can be incorrect as well and hence we refer to this dataset as noisy labeled dataset.

Once we have noisy labeled query data we assume that beside the observed query x and corresponding noisy label $\hat{y}$, we have two discrete latent variables — y and z that represent the true label and the label noise type, associated with query x respectively. The label noise type z is an 1-of-3 binary random variable, similar to that used in Xiao et al. [17], and is associated with the following intuitive meaning:

- The label is noise free, i.e., $\hat{y} = y$.
- The label suffers from a pure random noise, i.e., label $\hat{y}$ takes any possible random value other than the true label y.
- The label suffers from a confusing noise between clusters of overlapping classes. For example, languages like English and German may have overlapping vocabulary due to which the model may find it difficult to assign a particular class with high confidence.

Noise-type labels can, therefore, be interpreted with respect to the level of confidence shown by different labellers in label assignements to that query. Drawing motivation from above interpreration, we use the following conditions to assign noise-type label z to a query:

$$z = \begin{cases} 1 \text{ (noise-free)}, & \text{if } l_{h_1}(x) = \cdots = l_{h_m}(x) \\ 2 \text{ (random-noise))}, & \delta_1 \leq f(h_1(x)) \leq \delta_2 \\ 3 \text{ (confusing-noise)}, & f(h_1(x)) \leq \delta_1 \end{cases} \tag{2}$$

where δ_1 and δ_2 are hyper-parameters and used as thresholds; and $f(h_1(x))$ represents the predcton confidence with which labellng function $h_1(x)$ labels a query x.

In Section 3.3, we use the information from weak labels and noise-type labels to train a deep learning framework that help us filter queries with noisy labels from the dataset and approximate true label y.

3.2 Refining Noisy Label Data

Algorithm 1 briefly summarizes the steps for refinement of label noise data from training dataset in our framework. Using above-obtained noisy labeled dataset, we trained a Convolutional Neural Network (CNN$_1$) to compute class posteriors $p(y/x)$. This CNN can thus be treated as a standalone language identification model built upon weak labels computed in Section 3.1. Stochastic Gradient Ascent with backpropagation technique is used to optimize the weights of the network. The predictions of this classifier are fed in as pseudo-labels ($\tilde{y}$) for training of subsequent stages.

We trained two CNNs- CNN$_2$ and CNN$_3$, that compute the conditionals $p(y|\tilde{y}, x)$ and $p(z|\tilde{y}, x)$ using output $\tilde{y}$ from previous stage, CNN$_1$. While the class label probability distribution $p(y|\tilde{y}, x)$ is apparent, the intuitive meaning of $p(z|\tilde{y}, x)$ needs extra clarification. The quantity represents how likely a query is to be mislabeled given the predictions from the language identification model (CNN$_1$). Compared to traditional self-training approaches, under our approach output from previous itertaion is directly fed into next iteration. Computations of $p(z|\tilde{y}, x)$ hence ensure that model considers the assumption that an assigned label can be noisy and incorrect. The model is trained iteratively until either convergence is achieved or we start observing degradation of results on a held-out validation set.

3.3 Label Noise Pruning

From an intuitive point-of-view, the above setting provides us with two clues about each query in the unlabeled dataset: 1) what are the true labels for the query, and 2) how likely the query is to be mislabeled. In a typical ecommerce search system, unlabeled

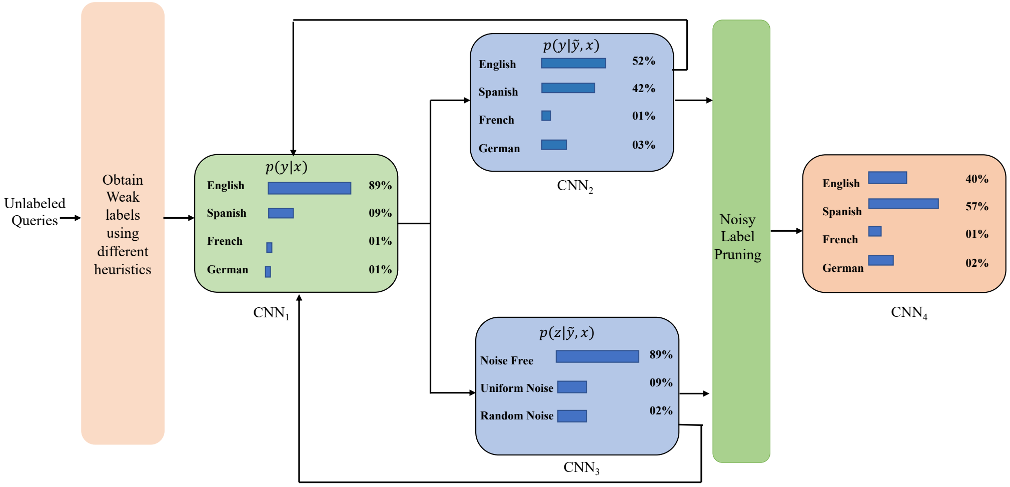


Figure 2: Flow Chart describing the proposed methodology. CNN$_1$ is used to predict the class label $p(y|x)$. Subsequent two CNNs compute the conditional probablity estimates of label and corresponding noise-type. Finally, queries with high probability of having 'Uniform-noise' and 'Random-noise' are pruned from training set. The final query language identification model is trained as CNN$_4$ with clean labeled data only.

data is abundant, and we can afford to discard queries with noisy predicted labels from our augmented training datasets. Queries in the unlabeled training dataset that are recognized as having noise-free labels using CNN$_3$ are retained. Such a filtered out dataset is then used to train the final query language identification model CNN$_4$. At inference time, the final trained model, CNN$_4$, with its computed weights, is used to predict the class of a query.

4 EXPERIMENTAL RESULTS

In this section, we compare our proposed approach against state-of-the-art text classification algorithms. We report our results for four languages - English, Spanish, French, and German. To begin with, we first describe training and validation (Section 4.1) datasets. Then we describe baseline systems (Section 4.2). Finally we present recall values for various precision points (Section 4.4).

4.1 Datasets

For training data preparation, we fetched 20 Million search queries issued in the last 4 months on a popular e-commerce search system. These queries were explicitly sampled from English, Spanish, French, and German interfaces of the search system. We pruned queries to eliminate less frequently occurring queries, and used the remaining 5 Million queries as our training dataset.

To generate an evaluation set, we randomly sampled unique search queries along with their frequency of occurrence from English, Spanish, French, and German interface. We used the Weighted Random Sampling algorithm [5] to sample 100 search queries from each interface randomly. The resulting 400 search queries were manually annotated to identify their actual language. Finally, our evaluation data had English, Spanish, French, and German queries in the ratio of 5:1:1:1 (254 English, 47 Spanish, 41 French, and 57 German queries).

4.2 Benchmark models and Baseline

In this section, we describe the specifics of our model and the FastText-based baseline model. In addition, we compare our proposed system trained on the weak and noisy label data with three different fastText models, and these are trained on three different datasets.

(1) *Generic Fasttext Model trained with Sentence Dataset* [BASELINE]: We used Tateoba sentence dataset to train generic multi-lingual identification model. The dataset contains labeled sentences belonging to 150 languages. We filtered out sentences belonging to English, Spanish, French, and German and used the resulting dataset for training the FastText model [2].

(2) *Query-Specific Fasttext model trained with High-confidence data*: We used the Generic Fasttext Model to make predictions on our training dataset. We built a subset of those queries where the associated classes were predicted with a high confidence score (≥ 0.8). Such a subset was then used to train a FastText model for carrying out the 4-class classification.

(3) *Query-Specific Fasttext model trained on Weakly supervised data*: We used multiple labeling functions (described in Section 4.3) to generate predictions on our training dataset and combined weak labels using the procedure described in Equation (1). We then trained a new FastText model with these refined labels.

Algorithm 1: Algorithm for training a language identification model based on weak supervision with noisy labels pruning.

```
Input  :
$X = \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$: Unlabeled query set
$h_1, h_2, \ldots, h_m$: labelling functions based on different heuristics
$\delta_1, \delta_2$: hyper-parameter threshold values to determine noise-type labels
Output:
$W$: Weight matrix corresponding to trained classification model
1  for $k = 1$ to $m$ do
2      Label each data point using labelling function $h_k$.
3  end for
4  Combine the labels obtained in Step 1-3 to get a set of weak label for each data point using Eq. (1). Lets
     $\chi_1 = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\}$ be the resulting data.
5  while convergence is not acheived do
6      Obtain noise-type label for each data point using Eq. (2) .
         Let $\zeta_1 = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\}$ be the result.
7      Train CNN$_1$ to estimate $p(y/x)$ probablities. Lets call its prediction output $\tilde{y}$.
8      Append predictions $\tilde{y}$ from CNN$_1$ to input queries as an additional feature to obtain input dataset for
         subsequent stages. Lets call the dataset $\zeta_1$ and $\zeta_2$.
9      Train CNN$_2$ using $\zeta_1$ to estimate probablities $p(y|\tilde{y}, x)$
10     Train CNN$_3$ using $\zeta_2$ to estimate probablities $p(z|\tilde{y}, x)$
11 end while
12 Filter out the datapoints $x_i$ from dataset $\chi$ corresponding to which $p(z|x_i) >= \delta$ for random-noise and
     confusing-noise classes. Let $D$ denote the obtained refined noise-free labeled training dataset.
13 Train final language identification model CNN$_4$ using dataset $D$.
```

(4) *CNN model trained on Weakly supervised data:* : In order to analyze the impact of noisy data pruning on model generalization, we trained a CNN model using approach discussed in Section 3.1 on weakly-supervised data.

(5) *Query-specific model trained on Noise-type labels and Weakly supervised data (Proposed Methodology)*: We fed our training dataset with refined labels into our proposed framework, as described through Equations (1) and (2). These noise type labels were then used to train a CNN as described in Section 3 and Figure 2. For each CNN, we used a ReLU activation function, filter windows of sizes 3, 4 and 5 with 100 feature maps each, a learning rate of 0.01, a dropout rate of 0.5, and a mini-batch size of 64. Our results were not sensitive to the choice of these settings.

4.3 Labeling Functions

To obtain weak labels for models (3) and (4) above, we combined the following three labeling functions:

- h_1: predictions from Query-Specific Fasttext model trained on High-confidence dataset
- h_2: outputs of detect_langs() of lang_detect[3] python library. Note that h_2 abstain from labeling when the predicted language is other than {English, Spanish, French, German}.
- h_3: we assigned query labels based on the majority of times they appear in a particular language interface. For example, if a query Q occurs two times or more frequently in the English interface than other language interfaces, then it is assigned with an English label.

The weak labels obtained using the above heuristics are combined to obtain a single set of probabilistic labels using Equation (1). The training data combined with these labels are used to train CNN$_1$ (refer to Figure 2). For training in the subsequent stage, the predictions $\tilde{y}$ obtained from CNN$_1$ is appended to queries, and this transformed dataset to train CNN_2 and CNN_3. The ground truth labels for CNN$_3$ optimized for predicting the label-noise type are obtained using heuristic defined in Eq. (2). We combined the prediction output of CNN$_2$ and CNN$_3$ to prune noisy labeled examples from the training dataset. Finally, only noise-free examples are used for training the final query language identification through CNN$_4$.

4.4 Results

Table 1 reports the relative improvements in the performance across various approaches and languages with respect to the Generic FastText model. We select the Generic Fasttext model as a baseline because it was trained on the standard language identification datasets, and not specifically on query text. However, we note that such a baseline underperforms with respect to other alternative models that we built. We posit that this difference is due to the difference in the nature of actual search queries versus sentences from the Taetoba dataset. Among the benchmarked models, we observe a significant lift in recall for all languages for models trained with weak supervision compared to those trained on just High-Confidence predictions of the Generic Fasttext model. Our proposed model is able to achieve equal or better performance in 7 (out of 12) combinations we tested for, while the next best performing models, namely, FastText + Weak-supervision and CNN + Weak-supervision were able to achieve equal or better performance in 4 (out of 12) combinations.

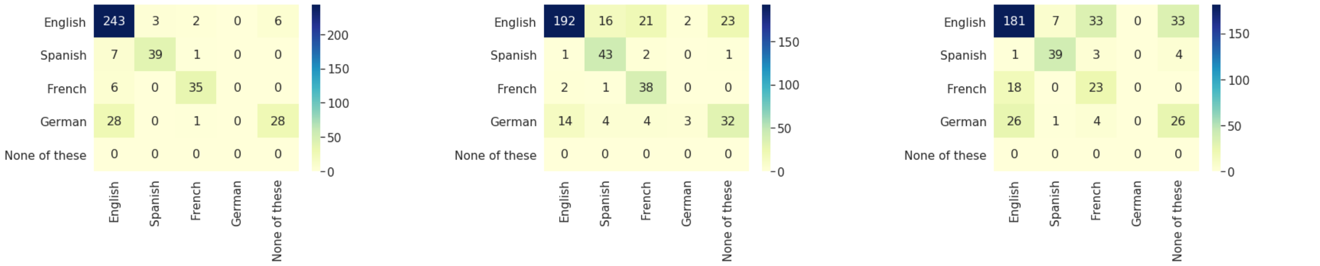
5 ERROR ANALYSIS

To further analyze the performance of our proposed methodology, we examined the misclassified queries from the test dataset. In general, we expect our models to confuse between languages that are closely related to one another. Such a hypothesis is confirmed by the confusion matrix presented in Figure 3. The confusion between these languages can be attributed to the shared character set and overlapping vocabulary of these languages. It can also be observed from Figure 3 that the proposed noisy model framework is able to better discriminate between languages than other methods.

[3] https://pypi.org/project/langdetect/

Model	Language	Recall @0.8Precision	Recall @0.85Precision	Recall @0.9Precision
Fasttext + High-confidence	English	-0.244	-0.354	-0.633
	Spanish	0.149	0.149	0.234
	French	-0.073	-0.024	-0.273
	German	0.123	0.00000	-
Fasttext + Weak-supervision	English	0.0118	0.0315	0.028
	Spanish	**0.170**	**0.170**	0.234
	French	**0.098**	**0.171**	0.098
	German	0.123	0.246	-
CNN + Weak-supervision	English	**0.020**	**0.053**	**0.068**
	Spanish	**0.170**	0.106	0.212
	French	-0.170	-0.098	-0.122
	German	0.093	0.182	-
Proposed Methodology	English	**0.020**	0.044	0.023
	Spanish	**0.170**	0.149	**0.255**
	French	**0.098**	0.146	**0.195**
	German	**0.171**	**0.263**	-

Table 1: Lifts in performance of language identification models in comparison to the baseline model.



	English	Spanish	French	German	None of these
English	243	3	2	0	6
Spanish	7	39	1	0	0
French	6	0	35	0	0
German	28	0	1	0	28
None of these	0	0	0	0	0

(a) Proposed Methodology

	English	Spanish	French	German	None of these
English	192	16	21	2	23
Spanish	1	43	2	0	1
French	2	1	38	0	0
German	14	4	4	3	32
None of these	0	0	0	0	0

(b) Fasttext + Weak Supervision

	English	Spanish	French	German	None of these
English	181	7	33	0	33
Spanish	1	39	3	0	4
French	18	0	23	0	0
German	26	1	4	0	26
None of these	0	0	0	0	0

(c) Fasttext + High-confidence

Figure 3: Comparision of confusion matrix of proposed method against other compared approaches on evalaution test data

On a detailed analysis of search queries, we found product names, which should ideally be treated as language-agonistic phrases, frequently appear in search queries (for example *maybelline eyebrow sombra*). This leads to increased confusion when mixed with other language keywords. Similarly, code-mix search queries containing non-local entity names such as model or brand name in a search query lead to erroneous results. For example, *pneus carlisle* (French → misclassified to English) gets incorrectly classified to English, because of presence of English brand name 'carlisle' and french word 'pneus' is completely ignored.

We also found smaller length queries, for example *kobo*, (English → misclassified to French), *asus* (English → misclassified to Spanish), *dewalt* (English → misclassified to German) are highly likely to get misclassified than longer length queries. This can be attributed to the fact that smaller length queries provide minimal context for the model to identify the language.

Finally, queries which contain digits or spelling errors including omitted spaces seemed to deteroirate model's performance. For example, *luminothrapie* (French → misclassified to English). Although such an effect is less pronounced in longer queries, where for example,*salomon femmesalomon 394671 femme canada* (French), this impact is compensated by presence of better contextual information.

6 CONCLUSION

In this paper, we proposed a learning framework that combines weak supervision with pruning of labels identified as being noisy on an unlabeled dataset for performing query language identification. Our framework segregates labels on examples as being noise-free, random-noise, and confusing-noise bins, with only noise-free examples being used for training the final query language identification model. Our resulting self-learning-based CNN setup demonstrated a higher performance than FastText and FastText combined with weak supervision on the same unlabeled dataset. Our error analysis indicates that we can further improve performance by augmenting training dataset to include mis-spelt examples including code mix queries with non-local entity names.

REFERENCES

[1] S. H. Bach, D. Rodriguez, Y. Liu, C. Luo, H. Shao, C. Xia, S. Sen, A. Ratner, B. Hancock, H. Alborzi, et al. Snorkel drybell: A case study in deploying weak supervision at industrial scale. In *Proceedings of the 2019 International Conference on Management of Data*, pages 362–375, 2019.

[2] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, dec 2017.

[3] J. Chen, Y. Hu, J. Liu, Y. Xiao, and H. Jiang. Deep short text classification with knowledge powered attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6252–6259, 2019.

[4] A. Conneau, G. Lample, M. Ranzato, L. Denoyer, and H. Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

[5] P. S. Efraimidis and P. G. Spirakis. Weighted random sampling with a reservoir. *Information Processing Letters*, 97(5):181–185, mar 2006.

[6] T. Jauhiainen, M. Lui, M. Zampieri, T. Baldwin, and K. Lindén. Automatic language identification in texts: A survey. *Journal of Artificial Intelligence Research*, 65:675–782, 2019.

[7] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[8] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao. Deep learning based text classification: A comprehensive review. *arXiv preprint arXiv:2004.03705*, 2020.

[9] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.

[10] N. Shuyo. Language detection library for java, 2010.

[11] G. Song, Y. Ye, X. Du, X. Huang, and S. Bie. Short text classification: A survey. *Journal of multimedia*, 9(5):635, 2014.

[12] H. Song, M. Kim, D. Park, and J.-G. Lee. Learning from noisy labels with deep neural networks: A survey. *arXiv preprint arXiv:2007.08199*, 2020.

[13] R. Tambi, A. Kale, and T. H. King. Search query language identification using weak labeling. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3520–3527, 2020.

[14] M. Toftrup, S. A. Sørensen, M. R. Ciosici, and I. Assent. A reproduction of apple's bi-directional lstm models for language identification in short strings. *arXiv preprint arXiv:2102.06282*, 2021.

[15] P. Varma and C. Ré. Snuba: automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 12, page 223. NIH Public Access, 2018.

[16] J. Wang, Z. Wang, D. Zhang, and J. Yan. Combining knowledge with deep convolutional neural networks for short text classification. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 2915–2921, 2017.

[17] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699, 2015.