Domain-specific transformer models for query translation

Mandar Kulkarni, Nikesh Garera, Anusua Trivedi
Flipkart Data Science
(mandar.kulkarni, nikesh.garera, anusua.trivedi)@flipkart.com

Abstract

Due to the democratization of e-commerce, many product companies are listing their goods for online shopping. For periodic buying within a domain such as Grocery, consumers are generally inclined to buy certain brands of products. Due to a large non-English speaking population in India, we observe a significant percentage of code-mix Hinglish search queries e.g., sasta atta. An intuitive approach to dealing with code-mix queries is to train an encoder-decoder model to translate the query to English to perform the search. However, the problem becomes non-trivial when the brand names themselves have Hinglish names and possibly have a literal English translation. In such queries, only the context (non-brand name) Hinglish words needs to be translated. In this paper, we propose a simple yet effective modification to the transformer training to preserve/correct Grocery brand names in the output while selectively translating the context words. To achieve this, we use an additional dataset of popular Grocery brand names. Brand names are added as tokens to the model vocabulary, and the token embeddings are randomly initialized. Further, we introduce a Brand loss in training the translation model. Brand loss is a cross entropy loss computed using a denoising auto-encoder objective with brand name data. We warm-start the training from a public pre-trained checkpoint (such as BART/T5) and further adapt it for query translation using the domain data. The proposed model is generic and can be used with English as well as code-mix Hinglish queries alleviating the need for language detection. To reduce the latency of the model for the production deployment, we use knowledge distillation and quantization. Experimental evaluation indicates that the proposed approach improves translation results by preserving/correcting English/Hinglish brand names. After positive results with A/B testing, the model is currently deployed in production.

1 Introduction

Due to the democratization of e-commerce, online shopping has evolved in recent times, where most customers choose to shop online. As an effect, the majority of product companies are keen on making their products available for online shopping. When it comes to domains such as Grocery, where users have to shop periodically, they typically have a preference for buying products of certain brands. Hence, for Grocery, it was observed that a significant portion of search queries contain brand names. Due to a large non-English-speaking population in India, we observe a significant percentage of code-mix Hinglish search queries. A Hinglish query is where one or more Hindi words are written in English, e.g., sasta atta. Since there are no standard spellings, we observe a large variation in the Hinglish words. We also observe many queries where brand names are misspelled.

An intuitive approach to deal with code-mix queries is to train an encoder-decoder model to translate the query to English and use an English search API to retrieve the products (Kulkarni et al., 2022). However, the problem becomes more challenging when the brand names themselves are Hinglish words and possibly have a valid English translation. We observe that in the Grocery domain, many brand names have Hinglish names, e.g. aashirvaad, gowardhan, veer, navratna etc. In such queries, only the context (non-brand name) Hinglish words need to be translated, and brand names (though Hinglish) must not be altered in the translation. E.g. for the query,

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 5: Industry Track, pages 89–95

'sasta dabur lal tel', a literal translation would be 'cheap dabur red oil'. However, the expected translation is 'cheap dabur lal oil' since 'dabur lal' is a brand name. Although most of the words in the query are Hinglish, only the first and last words need to be translated. If a brand name gets altered during the translation, it will lead to non-ideal search results. In some cases, the query does not need a translation even though it contains a Hinglish brand name, e.g., veer brand oil. If an English/Hinglish brand name is misspelled, it needs to be corrected in the translation. In general, the seq2seq model should be able to handle the following scenarios.

- the query has only English words with no spell errors: the model should output the query as it is
- the query has only English words with spell errors in either brand names or context words: the model should only correct the spell errors
- the query contains Hinglish words without brand names: the model should translate all Hinglish words to English
- the query contains Hinglish words with brand names: the model should selectively translate the Hinglish words without altering brand names. It should correct the brand names if it is misspelled.

To ensure such behavior, one would need large manually labeled data inclusive of many brand names. In this paper, we propose a simple yet effective modification to the transformer training to preserve/correct brand names in the output while selectively translating the context words. To achieve this, we use an additional dataset of high-demand Grocery brand names provided by the product team. First, to output brand names as a whole, we add them as tokens to the model vocabulary and randomly initialize the corresponding token embeddings. Further, we introduce a brand loss for training the translation model. Brand loss is a cross entropy loss computed using a denoising auto-encoder objective with brand name data. We warm-start the training from a generic pre-trained checkpoint (such as BART/T5) and further adapt it for query translation using the domain data. Results indicate that introducing brand loss significantly improves accuracy by preserving/correcting brand names in the translation. We also verify that introducing brand information as the loss is more effective than introducing it as the training data. The model is generic and can be used with English as well as code-mix Hinglish queries, alleviating the need for language detection. Further, to reduce the latency of the model for the production use-case, we use knowledge distillation and quantization. Using a large model as the teacher, we obtain pseudo-labels for a large set of unlabeled queries. We then train a small student opennmt (Klein et al., 2017) model on this dataset. We are able to achieve more than **28x** reduction in the latency with a slight drop in accuracy. Experimental results demonstrate the efficacy of the proposed approach.

2 Related works

Transformers (Vaswani et al., 2017) is the current state-of-the-art model for translation. Large-scale self-supervised pre-training of encoder-decoder models followed by domain-specific fine-tuning can significantly improve the translation quality with a limited labeled set (Lewis et al., 2019) (Raffel et al., 2020).

Search query translation is essential for Cross-Lingual Information Retrieval (CLIR). Bhattacharya et al. (Bhattacharya et al., 2016) use word vector emebedding and clustering to find groups of words representing the same concept from different languages. These multilingual word clusters are then used to perform query translation for CLIR between English, Hindi and Bengali. Kulkarni et al. (Kulkarni and Garera, 2022) proposes an approach to perform vernacular query translation without using any parallel corpus. Authors only utilize unlabeled query corpus from two languages, a pre-trained multilingual translation model, and train it with cross-language training to translate vernacular search queries to English. For code mix query translation, multilingual and English pre-trained encoder-decoder models have been explored (Jawahar et al., 2021) (Kulkarni et al., 2022). Kumar et al. (Kumar et al., 2020) explored statistical and neural ma-

Query	Ground truth	Without Brand loss	With Brand loss
asribad ata	aashirvaad atta	ashirwad atta	**aashirvaad** atta
dabber lal tel	dabur lal oil	dabber lal oil	**dabur lal** oil
emni rice brand oil	emami rice bran oil	emni rice bran oil	**emami** rice bran oil
daadis peanut khakra	daadi's peanut khakhra	grapes peanut seeds	**daadi's** peanut khakra
goverdhan desi ghee	gowardhan desi ghee	goverdhan desi ghee	**gowardhan** desi ghee
detol original	dettol original	original detol	**dettol** original
farnely all product	farmley all product	free all product	**farmley** all product
veer brand oil	veer brand oil	mustard oil	**veer** brand oil
all out mosquito refill	all out mosquito refill	eri mosquito refill	**all out** mosquito refill
ice cream kwality walls	ice cream kwality walls	ice cream kwality	ice cream **kwality walls**

Table 1: Effect of brand loss on accuracy. With the brand loss, the model preserves/corrects brand names and provides translation better aligned with the ground truth. Brand names are highlighted in boldface.

chine translation models for generating natural language questions from a given keyword-based query.

Few techniques have been explored to preserve some of the input tokens as it is in output. CopyNet (Gu et al., 2016) enables selective use of generate and copy mode. In the copy mode, an RNN-based model can choose sub-sequences from the input sequence to put them at appropriate places in the output sequence. While in generate mode, the model can generate new tokens. On similar lines, See et al. (See et al., 2017) proposed a hybrid pointer-generator network-based approach with an ability to copy words from input to the output while retaining the ability to produce novel words through the generator.

In contrast to these approaches, we enforce the model to copy brand names using an additional loss component computed on the brand name data. The model still has a default generate ability which helps in correcting misspelled brand names.

3 Proposed Approach

In the following sections, we provide details of the dataset and training methods.

3.1 Dataset

We use a manually tagged dataset for training the model. We have a total of ~116k manually tagged query set, which contains Hinglish as well as English queries. To make use of previously tagged queries, the dataset consists of queries from Grocery and other domains such as fashion, mobile, footwear, etc. From this, we use randomly chosen 5k samples as the validation set and ~111k for the training. We use a list of 2226 high-demand Grocery brand names to compute the brand loss. The list was provided by the product team. As the test dataset, we use 10715 manually tagged queries from the Grocery domain.

3.2 Training details

For training the translation model, we make two modifications as follows. First, we add a list of high-demand brand names as tokens in the model vocabulary and randomly initialize the corresponding token embeddings. Brand names are converted to lowercase before adding to vocab. This ensures that when a brand name is outputted in the translation, it would be outputted as a single entity, avoiding incorrect brand name variations.

We introduce a brand-specific loss in the model training. The translation model is trained with a combination of three loss components as follows.

$$L = l_{Supervised} + l_{DataAug} + \lambda\, l_{Brand} \qquad (1)$$

where λ indicates the weighting factor for the brand loss. $l_{Supervised}$ indicates the standard cross entropy loss with parallel corpus. $l_{DataAug}$ indicates the loss calculated with spell and auto-encoder data augmentations as described in section 3.3.

For calculating l_{Brand}, we use cross-entropy loss with denoising autoencoder objective

with brand name data using simple CharDrop data augmentation. Since non-English speakers attempt to spell the words based on the phoneme sound of it, we noticed that typically the first and last character of the brand is spelled correctly while the spelling mistakes are present in the middle of the word. To emulate this, we randomly drop a character from 30-50% of the brand name words and use original brand names as the target. Following are some of the brand name training examples.

Noisy Brand name	Target
asirwaad	ashirwaad
milky freh	milky fresh
dabur vaika	dabur vatika

Table 2: Brand name augmentations

l_{Brand} is computed with the teacher forcing technique. We set λ to 1 for all experiments. We also experimented by increasing and decreasing the value of λ, however, it did not lead to any significant change in the accuracy.

We use a pre-trained BART-base model to warm-start the training and fine-tune it further on the manually tagged data. The model is fine-tuned using AdamW optimizer with a learning rate of 1e-5 and batch size of 16. The model is trained till the validation loss does not improve for three consecutive epochs. We use label smoothing (Vaswani et al., 2017) during the training, where we set the label smoothing parameter to 0.1 for all the experiments. We use beam search decoding during the inference, where the beam size is set to 3. The model has ~141M trainable parameters post adding the brand tokens.

3.3 Data Augmentations

We experimented with Autoencoder and spell augmentation to compute data augmentation loss ($l_{DataAug}$). For Autoencoder, we use target English text as the input and train the model to reconstruct it. Though simple, it has shown to be effective in query translation since it provides an advantage similar to a language model regularizer (Kulkarni et al., 2022). For the batch of labeled queries, we add spell augmentations to the source (Ma, 2019) and train the model with the same target. For each batch of queries, data augmentation is chosen randomly.

Setting	BLEU
With Brand loss	**70.9**
Without Brand loss	68.8

Table 3: BLEU score comparison result

4 Results

Table 3 shows the BLEU score comparison of different model settings on the test set. In the first experiment, we verify the effectiveness of additional brand loss during the training. We train the model with and without brand loss. From the BLEU score comparison, it can be seen that brand loss training provides good improvements in test accuracy. In table 1, we show the comparison of query translation results with and without brand loss. With the brand loss, the model corrects the brand names whenever it is entered wrongly (first 7 examples). It also preserves brand names better when it's entered correctly (last 3 examples). Overall, the model provides translations better aligned with the ground truth.

4.1 Using brand names as data

Intuitively, it's possible to input the brand name information as the parallel corpus, where we can add CharDrop augmentation to the brand names, and the original brand name can be used as the target. Hence, we wanted to verify the effectiveness of introducing brand information through the loss compared to inputting it through the training data. We created additional training data from the brand names with CharDrop augmentations and appended it to the original training set. We use 50 augmentations for each brand name. Table 5 shows the BLEU score comparison result. We notice that adding brand info as a loss is more effective than adding it as training data. This could be because, with the brand as loss, the model is able to translate context words more effectively. Table 4 shows the query translation comparison result. Note that brand as loss is better at correcting misspelled brand names while providing better translations of context words.

Query	Ground truth	Brand as data	Brand as loss
nabrtan tel	navratna oil	olive oil	navratna oil
cubes spice masala	cubes spice masala	cake spice masala	cubes spice masala
fitme ka face pauder	fit me face powder	face powder offitme	fit me face powder
dabur gulab jal 1 litre	dabur gulab jal 1 litre	dabur rose water 1 litre	dabur gulab jal 1 litre
colgeat charcol offer	colgate charcoal offer	coffee charcol offer	colgate charcoal offer
boork bond taja tea	brooke bond taaza tea	boork bond tea	brooke bond taaza tea
fiama soap all mox	fiama soap all mix	fiama soap all mox	fiama soap all mix
kesar ka sabudana	kesar sabudana	saffron seeds	kesar sabudana

Table 4: Comparison result for inputting brand information as loss vs inputting through training data. Note that with the brand as a loss, context words are better translated.

Setting	BLEU
Brand info as data	69.6
Brand info as loss	70.9

Table 5: BLEU score comparison for brand as loss vs brand as data

4.2 Comparison with T5

We compared the results of BART-base with T5-base and T5-small models under similar training settings, i.e., adding brand tokens to the vocab and training with brand loss. Table 6 shows the comparison result. We noticed that BART works significantly better as compared to T5. This could be because denoising training objectives such as brand loss and data augmentation are more aligned with the BART pre-training than T5. Hence, BART can provide good results with a limited labeled set, especially when brand token embeddings need to be learned from scratch.

Setting	BLEU
T5-base	59.8
T5-small	57.2
BART-base	70.9

Table 6: Comparison with T5 model

4.3 Pre-training on large query

Since the search model would be witnessing large traffic and a variety of queries, we pretrain BART-base model on a large query parallel corpus to make it suitable for production use case. We collected a large Hindi (Devanagari) unlabeled query corpus from the internal database. Since our Hindi search model currently supports different verticals such as fashion, mobile, footwear, etc., we suspect only a small percentage of Grocery related queries in the dataset. The Hindi queries are detected using a simple script-based detection. If any of the characters in the query are from Devanagari unicode range, the query is termed Hindi. We then use an in-house Hindi to English query translation model to create a parallel corpus from the unlabeled set. Further, we use an in-house *transliteration* model to convert a Hindi query to a Hinglish query. This way, we obtained a ~38M Hinglish to English query parallel corpus for training. The model is trained using AdamW optimizer with a learning rate of 5e-6. We pre-trained the BART-base model on this large set and then finetuned on the manually tagged set in the same manner described in section 3.2. Table 7 shows the result of the experiment. Pre-training on the large set gives a significant boost to accuracy. To verify if brand loss based finetuning still complements the advantage provided by the pre-training, we finetuned the query pretrained model without the brand loss. It can be seen that training with brand loss boosts accuracy in addition to the pre-training.

Query-pretraining	Brand loss	BLEU
Included	Included	**73.1**
Not included	Included	70.9
Included	Not included	71.5

Table 7: Effect of large scale query pre-training

5 Knowledge distillation for improved latency

The search query translation models are user-facing and need to have low latency to support high throughput. Though the BART-base model with query pre-training and fine-tuning provided good accuracy on the test set, it was not sufficient for production deployment due to the latency constraints. We observed that the p95 latency of the BART-base model with PyTorch implementation was ~200 ms, which is not acceptable for the production use-case.

To reduce the latency of the model, we use knowledge distillation with open-nmt (Klein et al., 2017) framework. Open-nmt provides a Ctranslate wrapper for faster inference, making it a good choice for low latency use-cases. Our approach is to train a small open-nmt student model using Grocery BART-base model as the teacher model. Since the student model resides in another programming framework, we use a pseudo-labeling approach to transfer knowledge from the teacher to the student. To create a parallel corpus for open-nmt model training, we obtain translation labels on ~38M query set using the teacher model. We then train the open-nmt model on this large parallel corpus and the manually tagged set. We use a single layer open-nmt model with a vocab size of 18k and a hidden dimension of 384. The model has ~23M trainable parameters. For open-nmt model as well, we add the brand name tokens to the vocab. We use weight quantization during model inference. Table 8 shows the BLEU score comparison result with the open-nmt student model. The student model provides more than **28x** speed up for the inference with just a 0.2 drop in the BLEU score. The reason a single layer student model could be providing comparable results to the teacher model can be two-fold. First, search queries rarely have grammar and hence may not a deeper network for translation. Second, the teacher through pseudo labeling is providing cleaner and consistent labels for the student to learn from.

We performed A/B testing of the open-nmt student model w.r.t. an earlier model which does not use brand loss. We observed 10 basis points (bps) improvement in search Click-Through-Rate (CTR) and improved search conversion. The model is currently deployed in production and serves a large volume of queries.

Setting	BLEU	p95 latency
BART Teacher	73.1	~200 ms
open-nmt student	**72.9**	**~7 ms**

Table 8: Knowledge distillation with open-nmt

6 Conclusion

In this paper, we proposed a simple yet effective approach for domain-specific query translation. For the grocery domain, it was noticed that a significant percentage of queries contained brand names due to user preferences and periodic buying. We also observed a significant percentage of code-mix Hinglish queries and queries with grammatical errors. Since some grocery brand names are themselves Hinglish words, we wanted a brand-aware query translation model. To better preserve brand names in translation, we added brand name tokens to the model vocab and introduced an additional brand loss in transformer training. The modification improved translation accuracy by depicting desired brand name preserving effect. To reduce the latency of the model for the production deployment, we used knowledge distillation with the open-nmt student. Using a large model as a teacher and with pseudo labeling, we trained a single layer open-nmt student model. We could obtain more than a 28x reduction in latency with a slight drop in accuracy. After positive results with A/B testing, the model was deployed in production.

References

Paheli Bhattacharya, Pawan Goyal, and Sudeshna Sarkar. 2016. Query translation for cross-language information retrieval using multilingual word clusters. In *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP2016)*, pages 152–162, Osaka, Japan. The COLING 2016 Organizing Committee.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning.

Ganesh Jawahar, El Moatez Billah Nagoudi, Muhammad Abdul-Mageed, and Laks V. S. Lakshmanan. 2021. Exploring text-to-text transformers for english to hinglish machine translation with synthetic code-mixing.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada. Association for Computational Linguistics.

Mandar Kulkarni, Soumya Chennabasavaraj, and Nikesh Garera. 2022. Study of encoder-decoder architectures for code-mix search query translation.

Mandar Kulkarni and Nikesh Garera. 2022. Vernacular search query translation with unsupervised domain adaptation.

Adarsh Kumar, Sandipan Dandapat, and Sushil Chordia. 2020. Translating web search queries into natural language questions.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.