Modeling Across-Context Attention For Long-Tail Query Classification in E-commerce

Junhao Zhang
junhao.zjh@alibaba-inc.com
Alibaba Group
Hangzhou, China

Weidi Xu
weidi.xwd@antfin.com
Ant Financial Services Group
Hangzhou, China

Jianhui Ji
jianhui.jjh@alibaba-inc.com
Alibaba Group
Hangzhou, China

Xi Chen
gongda.cx@taobao.com
Alibaba Group
Hangzhou, China

Hongbo Deng
dhb167148@alibaba-inc.com
Alibaba Group
Hangzhou, China

Keping Yang
shaoyao@taobao.com
Alibaba Group
Hangzhou, China

ABSTRACT

Product query classification is the basic component for query understanding, which aims to classify the user queries into multiple categories under a predefined product category taxonomy for the E-commerce search engine. It is a challenging task due to the tremendous amount of product categories. And a slight modification to a query will change its corresponding categories entirely, e.g., appending the "button" to the query "shirt". The problem is more severe for the tail queries which lack enough supervision information from customers. Motivated by this phenomenon, this paper proposes to model the contrasting/similar relationships between such similar queries. Our framework is composed of a base model and an across-context attention module. The across-context attention module plays the role of deriving and extracting external information from these variant queries by predicting their categories. We conduct both offline and online experiments on the real-world E-commerce search engine. Experimental results demonstrate the effectiveness of our across-context attention module.

CCS CONCEPTS

• **Information systems** → **Information retrieval query processing**; • **Computing methodologies** → *Natural language processing*.

KEYWORDS

E-Commerce Search; Query Classification

ACM Reference Format:
Junhao Zhang, Weidi Xu, Jianhui Ji, Xi Chen, Hongbo Deng, and Keping Yang. 2021. Modeling Across-Context Attention For Long-Tail Query Classification in E-commerce. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining (WSDM '21), March 8–12, 2021, Virtual Event, Israel.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3437963.3441822

WSDM '21, March 8–12, 2021, Virtual Event, Israel

ACM ISBN 978-1-4503-8297-7/21/03…$15.00
https://doi.org/10.1145/3437963.3441822

1 INTRODUCTION

In E-commerce, the product category taxonomy is a fundamental framework. Sellers deliver products under one of the suitable leaf categories. To cover all kinds of products, the category taxonomy involves ten of thousands leaf categories. Query classification can be formulated as a multi-label classification task. Unfortunately, due to more than thousands leaf categories, manually-annotated data is both expensive and time-consuming. In the search engine, it is common to utilize the click-through data as implicit feedback signals [20, 30].

It's well known that search queries follow the power law distribution [31]. Infrequent queries are individually rare, but as a whole they possess a significant portion of the query volume. However, tail queries lack enough customer behaviors information, and hence cause difficulty in identifying related categories from click-through log data. It's common practice to build classifiers to learn from frequent queries [30].

Deep neural networks have been successfully applied to multi-label classification. A common approach called Binary Relevance [3, 6, 22, 26, 38] transforms it into multiple single-label classification problems, and predict each label independently without considering label dependencies. To capture co-occurrence among the labels, sequence generation models [7, 19, 27] use a decoder with the attention mechanism [2] to generate target labels sequentially, which predicts next label based on previously predicted labels.

Category intents exhibit across-context correlation among query and its variation, which is not yet studied. Figure 1 illustrates the idea of across-context attention. Top left part of the figure depicts across-context attention from query "table household" to its variant query "simple Mahjong table household". It pays most attention to "Mahjong". Top right part depicts that another query "Mahjong table" pays most attention to "Mahjong" and "table" of the same variant query. How can we understand a new query "Mahjong table household"? Is it similar with left query or right query? Bottom part of the figure depicts new query shares a similar attention distribution on variant query with right query "Mahjong table". We can conclude that the difference of key auxiliary information from variant queries shall help models to distinguish category intents of a query.

Inspired by these observations, in this paper, we propose to model the contrasting/similar relationships between similar queries for long-tail query classification in E-commerce. Our framework is

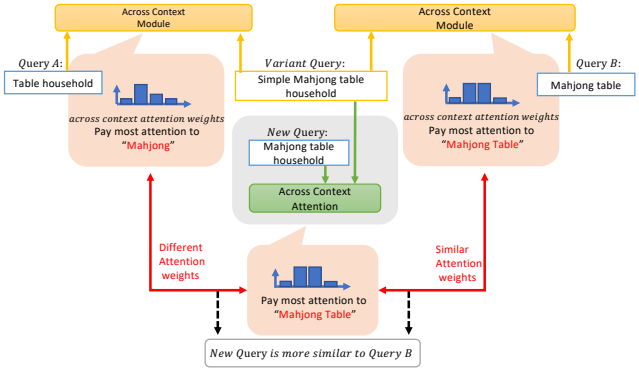


Figure 1: Across context attention is helpful to understand a new query.

composed of a base model and an across-context attention module. The query representation is learned not only from labeled categories but also from query variations. By predicting the categories of variant queries, our across-context attention module is capable to derive and extract external information from variant queries that benefits the identification of original query's category intents. Our across context module brings no extra computing cost during prediction. Offline evaluations on a large human labeled data-set from search log show that our module can improve base models of flexible query encoders. Online evaluations in a real-world e-commerce search engine of *www.taobao.com* demonstrate our across-context module is effective.

Our contributions are listed as follows:

- We propose a novel end-to-end trainable across-context attention module for deep-learning models.
- We investigate the performance of models with across-context attention module both quantitatively and qualitatively. Experimental results show that our proposed module improves base models.

The rest of the paper is organized as follows. We introduces the related work in Section 2. In Section 3, we describe our module and introduce it into base models. The methods of data collection, experiments and analysis are represented in Section 4. We conclude this paper and discuss future work in the last section.

2 RELATED WORK

2.1 Product Query Classification

Shen et al. [30] proposed two approaches of collecting training data for product query classification. One is obtaining the labels of queries from click-through log data using heuristic rules, and the other is translating labeled product titles to labeled pseudo query. The former votes relevant categories by the ratio of its clicked products over total number of clicked products. A large threshold is used to reduce the noise in data. The latter leverages frequently clicked pair of query and product title as a training data, and a tuple-ngram-based translation model is learned to translate pseudo query. The authors found that using pseudo queries as the training data can improve classification performance compare to using product title, but is worse than the directly collecting approach from click-through log data. Lin et al. [20] also proposed using implicit feedback from user click behavior as a signal to collect training data for query classification in e-commerce domain. In our work, we also use click-through log data as our training data. We train models on frequent queries since tail queries lack enough customer click through behavior.

2.2 Multi-Label Text Classification

Multi-Label Text Classification (MLTC) aims to predict multi labels for each text. An important group of method is problem transformation. Boutell et al. [4] train separate binary classifiers independently for each label, without consider the label dependency. Tsoumakas and Katakis [32] map MLTC to multiple single-label learning tasks, by combining label sets into single-label. In this way, label dependency is considered. Read et al. and Cheng et al. [8, 29] passing label dependency along a chain of classifiers. Li et al. [18] propose a joint learning algorithm for predictions-as-features methods, that allows the feedbacks to be propagated from the latter classifiers. Other works adapt machine learning models to solve multi-label problems directly. Elisseeff and Weston [10] propose a ranking support vector machine to handle multi-label problem. Zhang and Zhou [39] propose a nearest-neighbor based method, which finds k nearest neighbors in the training set for the unseen instance, then, maximum a posteriori principle is utilized to determine the final label set.

Recently, deep neural network models have been successfully applied to multi-label classification. Neural network models belonging to binary relevance demonstrate a certain degree of improvements [3, 26, 38]. To take label correlation into account, sequence to sequence models are applied to multi-label text classification [7, 19, 27, 36]. The basic network architecture is an semantics encoder to extract information from text, which is then passed to an RNN-based decoder to generate label sequences. Attention mechanism is applied to capture useful information from text for each decoder state. In our work, we take these kinds of neural network models as a base model, and propose an across-context attention module to help base model better understand the original query.

3 PROPOSED APPROACH

In this section, we introduce our across-context module in detail, and analyze the impact of the our across-context module during training and predicting.

3.1 Overview

Here, we define some notations and present our models for tail query classification in E-commerce. Denote N as the total number of leaf categories. Denote a query as $\mathbf{x}$, query classification aims to assign a subset $\mathbf{y}$ of categories from all leaf categories to $\mathbf{x}$. Here the size $|\mathbf{y}|$ of set is not fixed, and should equal to the number of relevant categories of query $\mathbf{x}$ if query classification performs perfect. In E-commerce, query and its variations may have similar or contrasting category intents. We denote a variant query of original query $\mathbf{x}$ as $\mathbf{z}$, it contains m words, and its relevant categories set in training data is $\mathbf{r}$.

An overview of our proposed models is shown in figure 2. It consists of two components, i.e., a base model and an across-context module. Base model takes query $\mathbf{x}$ as input, and encodes $\mathbf{x}$ into a hidden space, and then predicts categories $\mathbf{y}$. We denote the hidden representation of query $\mathbf{x}$ as $\mathbf{h}$. Our across-context module is an auxiliary task in training stage. To model the contrasting/similar category intents between literally similar queries $\mathbf{x}$ and $\mathbf{x}$', it utilizes across-context attention on variant query $\mathbf{z}$ to obtain auxiliary information. Queries with contrasting/similar auxiliary information shall have contrasting/similar query representations. As a result, the base model adjusts the parameters of query encoder to better represent a query.

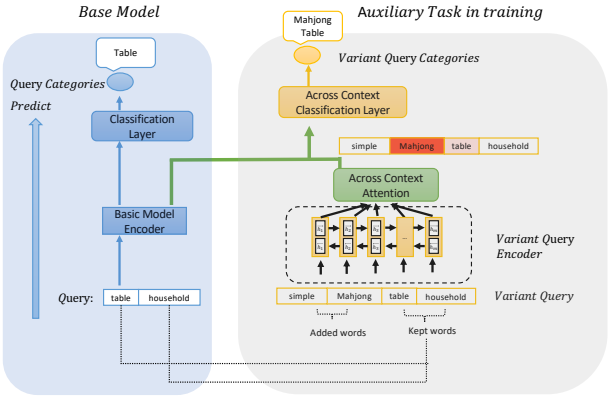


Figure 2: Across Context aware deep learning models for Query Classification in E-commerce.

3.2 Base Model

The query encoder is a part of our neural architecture. It takes query $\mathbf{x}$ as input, and produces query representation $\mathbf{h} \in \mathbb{R}^D$. Then an output layer with N units corresponding to the scores assigned to each category, denoted by $\mathbf{f} \in \mathbb{R}^N$.

$$\mathbf{f} = \mathbf{W_o h}\,, \tag{1}$$

where $\mathbf{W}_o \in \mathbb{R}^{N \times D}$ are weight matrices of the output layer, where each row of $\mathbf{W}_o$ can be viewed as the hidden representation of corresponding category.

Binary cross-entropy loss is suitable for multi-label classification task [26]. In our base model, we define the objective function for each sample as follows:

$$L_B = -\sum_{i=1}^{N} y_i log(\sigma(\mathbf{f}_i)) + (1 - y_i) log(1 - \sigma(\mathbf{f}_i))\,, \tag{2}$$

where $y_i = 1$ means category (i is in subset) $\mathbf{y}$.

Our base model can be any text classification neural network models [5, 12, 14, 15, 21, 23, 34, 37], that take the query as input, and produce a fixed dimensional embedding representation $\mathbf{h}$. For example, BiLSTM-based models use bidirectional LSTMs [11] to encode a source query, and then use a layer like max pooling across all LSTM hidden states to get a fixed-length query embedding vector. CNN models use same scheme, but substitute biLSTM with convolutional network [17].

Sequence generation models [19, 27, 36] can also be our base model, although they do not compress all the necessary information of a source query into a fixed-length vector. At each time step of the decoder, we take the input of the last output layer as our query representation $\mathbf{h}_t$. The decoder generates a conditional probability distribution over all categories at time step t as follows:

$$\mathbf{o}_t = \mathbf{W_o h}_t\,, \tag{3}$$

$$p\left(y_t | \, y_{<t}, \mathbf{x}\right) = softmax\left(\mathbf{o}_t + \mathbf{I}_t\right)\,, \tag{4}$$

where $\mathbf{W}_o \in \mathbb{R}^{N \times D}$ are weight matrices of output layer, and each row of $\mathbf{W}_o$ can be viewed as hidden representation of corresponding category, and $(\mathbf{I}_t)_i = -\infty$, if category i is previously generated, and $(\mathbf{I}_t)_i = 0$ otherwise, such that prevents decoder from generating repeating categories. The objective function for each sample is defined as follows:

$$L_B = -log(\prod_{t=1}^{|\mathbf{y}|} p\left(y_t | y_{<t}, \mathbf{x}\right))\,, \tag{5}$$

where y_t denotes one category belongs to $\mathbf{y}$, and $y_{<t}$ denotes $y_1, y_2, ..., y_{t-1}$.

3.3 Across Context Module

3.3.1 Variant query encoder. We employ a bidirectional Long Short-Term Memory [11] as our variant query encoder. Variant query encoder sequentially takes each word of variant query as input from two directions, and produces bidirectional hidden state for each word. We can formulate it as follow:

$$\begin{aligned} \overrightarrow{\mathbf{a}_i} &= \overrightarrow{LSTM}\left(\overrightarrow{\mathbf{a}_{i-1}}, \mathbf{z}_i\right)\,, \\ \overleftarrow{\mathbf{a}_i} &= \overleftarrow{LSTM}\left(\overleftarrow{\mathbf{a}_{i+1}}, \mathbf{z}_i\right)\,, \end{aligned} \tag{6}$$

where $\mathbf{z}_i$ and $\mathbf{a}_i$ denote the embedding vector and hidden representation of i-th word of variant query respectively.

3.3.2 Across context attention. By predicting categories of variant query, attention on variant query can provide original query with necessary auxiliary information and indicate category intents of original query from another perspective.

The across context attention can be formulated as follows:

$$e_i = \mathbf{v}_a^T tanh\left(\mathbf{W_a h} + \mathbf{U_a a}_i\right)\,, \tag{7}$$

$$\alpha_i = \frac{\exp\left(e_i\right)}{\sum_{j=1}^{m} \exp\left(e_j\right)}\,, \tag{8}$$

$$\mathbf{c} = \sum_{i=1}^{m} \alpha_i \mathbf{a}_i\,, \tag{9}$$

where $\mathbf{v}_a$,$\mathbf{W}_a$,$\mathbf{U}_a$ are weights parameters, and m is the number of words in variant query $\mathbf{z}$.

By predicting categories of variant query, across context vector $\mathbf{c}$ compresses all the necessary information from variant query $\mathbf{z}$ with respect to original query $\mathbf{x}$. The auxiliary task generates a conditional probability distribution over all categories, and can be formulated as follows:

$$\mathbf{d} = f\left(\mathbf{W_d h} + \mathbf{V_d c}\right) , \tag{10}$$

$$\mathbf{p}\left(r_i|\mathbf{x}, \mathbf{z}\right) = softmax\left(\mathbf{W_p d}\right) , \tag{11}$$

where $\mathbf{W}_d$,$\mathbf{V}_d$,$\mathbf{W}_p$ are weight parameters, $\mathbf{W}_p$ maps $\mathbf{d}$ to a N-dimensional vector. f is a non-linear activation function and $r_i \in \mathbf{r}$ is one category of variant query $\mathbf{z}$.

3.4 Training

The objective function of across context module can be jointly optimized with the objective function of base model under a multi-task framework. We formulate the extra objective function L_{AC} as follows:

$$L_{AC} = \frac{1}{|\mathbf{r}|} \sum_{i=1}^{|\mathbf{r}|} \log \mathbf{p}_{AC}\left(r_i|\mathbf{x}, \mathbf{z}\right) . \tag{12}$$

The final objective function of our model is a weighted sum of L_B and L_{AC}:

$$L = L_B + \lambda L_{AC} . \tag{13}$$

By optimizing the final objective function, both direct learning from labeled categories and contrastive learning from query variations tune query encoder of base model to produce better query's representation.

3.5 Model Inference

The previously proposed across context module can model the contrasting/similar relationships between similar queries. Base model is equipped with our module by adding an extra objective function in training procedure. Because only base model itself participates in predicting the categories of input queries, no extra computation is brought from our across-context module.

4 EXPERIMENTS

In this section, we describe our experiments in detail. We first present the training data and test data, as well as the evaluation metrics. Then we introduce our main results. After that, we analyse and discuss the effect of our proposed module, followed by a case study. Finally, we present online evaluation in a real-world e-commerce search engine.

4.1 Data

4.1.1 Training data. For our train data, we sample search queries and their corresponding click data from one year of search logs. We have frequency of click between pair of query (q) and category (c). We take logarithm of click frequency for each (q, c) , and normalize it by the maximum value under each query. We refer to the maximum normalized value as raw-click relevance. Only (q, c) that has a relevance larger than a threshold t is taken as our training data set. In practise, we found $t = 0.5$ is suitable to balance precision and recall. Moreover, only frequent queries with average daily page views larger than 20 are selected. We denote it as raw-click training dataset. Table 1 shows the statistics of our training data set.

For sequence generation models require category order, we sort categories under each query by the descending order of raw-click relevance.

Table 1: Summary of training dataset and test dataset. We present the total number of queries and total number of distinct categories in datasets, as well as the average number of words and related categories of a query.

Dataset	Query	Category	Avg words	Avg categories
training data	8489969	11610	2.86	4.04
test data	5880	5575	3.55	3.65

Table 2: Examples in testset. Value of 1 in last column means query and category is manually labeled as related.

Query	Category	relatedness
"nine years-old boy vest"	children's clothes»vest	1
"nine years-old boy vest"	sport»football»clothes	0
"nine years-old boy vest"	men's fashion»vest	0
"nine years-old boy vest"	children's clothes»T-shirt	0
"big fur collar down jacket"	women's fashion»down jacket	1
"big fur collar down jacket"	sport clothes»down jacket	1
"big fur collar down jacket"	outdoors clothes»down jacket	1
"big fur collar down jacket"	sport clothes»cotton coat	0

Besides, before feeding the training dataset into our models, we shall find queries with slight variations for each input query. Variant queries are also from training data, and literally include original query and have one more word for simplicity. Variant queries bring no additional knowledge from outside the training data, and training data provides the variant query categories. Around 33 percent of queries find its variants from training data set. For the remaining 67 percent of queries without variants, we use query itself as its variant for consistency.

4.1.2 Test data. For our test data, we sample 5880 infrequent queries from search log with average daily page views less than 20. Most infrequent queries are infrequent combination of frequent words in the real-world e-commerce search engine. For each query, we collect its candidate categories from the products it clicked in search log, as well as the recalled products by this query or its drop-word variation in search engine. Three human labelers are invited to label whether pair of query and its candidate category is relevant or not. We combine the labeling results from the three labelers by voting when the labelers disagree with each other for a certain query. Finally, 41324 pairs of query and category with labels indicate binary relevance form our test data, in which 21462 pairs are labeled as positive. Table 2 gives some positive and negative examples in our test data.

4.2 Evaluation Metrics

We use micro-precision, micro-recall, micro-F_1 and group-auc [40] as our evaluation metrics.

- Micro-precision: evaluates the fraction of total true positives over total positive instance-label pairs.
- Micro-recall: evaluates the fraction of total true positives over total true instance-label pairs.

- Micro-F_1: a weighted average of the micro-precision and micro-recall.
- Group-auc: aggregates all test data according to query, then calculates AUC results in each single group, and averages these weighted AUC.

4.3 Implementation Details

We set dimension of word vector to 200. We use one layer bi-LSTM as variant query encoder. Dimension of hidden states of variant query encoder is 400. Dimension of across context vector $\mathbf{c}$ is set to 200. And the combination of query representation and across context vector have 200 dimensions. Other hyper-parameters are set the same as base model. For DAN, CNN, BILSTM and SSA base models, we adopt Adam [16] optimization method to minimize the binary cross-entropy loss over the training data. For the hyper-parameters of the Adam optimizer, we set the learning rate 0.0005, two momentum parameters $\beta_1 = 0.9$ and $\beta_2 = 0.995$ respectively, $\epsilon = 1 \times 10^{-8}$.

For sequence generation models, we adopt adagrad [9] as our optimization method to update network parameters. Learning rate is set to 0.15, and the accumulator values in adagrad is initialized to 0.1. We clip the gradients [28] to the maximum norm of 2.0.

During training, we monitor model's micro-F1 score performance on the validation set, and the best one is selected and evaluated on the test set.

4.4 Baselines

We compare our proposed methods with the following baselines:

- Click: This method takes logarithm of 2-month click frequency between query and categories, and normalizes it by the maximum value under each query. It is zero when query did not click at a category at all.
- KNN [1]: This method classifies a test instance by identified its k nearest neighbors in the training set. We use tf-idf [13] as term weights to model the similarity between queries. Finally, all positive categories of its k-nearest neighbors are accepted as the categories of test query. It's a baseline to show whether lexically similar variants can help with better understanding of tail queries.
- SGM [36]: This model is an attention-based sequence to sequence model and introduces global embedding to relief exposure bias by considering all informative signals contained in already predicted categories.
- DAN [12]: Deep average network averages the input word vectors, and feeds it into two ReLU nonlinear layers [25] and then a final linear transformation layer with sigmoid activation.
- CNN [15]: Convolutional neural network model uses multiple convolution kernels to extract text features. A max-pooling layer transforms features into a fix-dimensional vector. We feed it into two ReLU nonlinear layers and then a final linear transformation layer with sigmoid activation.
- BILSTM [23]: bidirectional LSTM model uses a bidirectional LSTM in each direction, and uses max pooling across all LSTM hidden states to get the query embedding vector. We feed it into two ReLU nonlinear layers and then a final linear transformation layer with sigmoid activation.
- SSA [21]: Structured self attention model uses multiple hops of self attention on bidirectional LSTM to extract text features. We feed the feature representation to two ReLU nonlinear transformation layers, and then to a linear transformation layer with sigmoid activation. We set the hyper-parameters in a similar way as the work[21].

4.5 Main Results

Table 3 shows the performance of the proposed method on manually labeled test set. When equipped with our across-context module (AC), most base models achieve better results than its base version, which verifies the effectiveness of module, although it is still hard to recall relevant tail categories. Click method performs poor, due to lack of enough supervision information from customers. KNN method increases recall metric by a large margin compared with click method. It proves the value to learn from frequent queries. However, literally similar queries may have different category intents, which decreases precision metric. Since tail queries lack enough supervision information from customers, and different category intents exist among its variant queries, it is better to model frequent queries with AC module and then generalize the model to tail queries. SGM predicts a target label depends on both source query and predicted labels, in the case of focusing more on the fluency of label sequence, the predicted labels can mislead model to generate a category not faithful to source text and is adverse to precision metric, as noted in neural machine translation [33]. Also, beam search tends to favor shorter results over longer ones [35], which may cause the decrease of recall metric. In terms of precision metric, the improvements of SGM-AC against SGM is 7.2%. Remind that our module corrects the input vector of the last output layer at each time step of decoder, it avoids the misleading effect of predicted categories to some extent. DAN-AC does not improve much over DAN in terms of f1-score metric, but gain 0.9% improvement in terms of GAUC. We attribute it to the inflexibility of average embedding encoder. For base models with flexible query encoders like CNN, BILSTM and SSA, the improvements by using across-context module are 1.3%, 2.1%, 1.2% respectively in terms of f1-score metric.

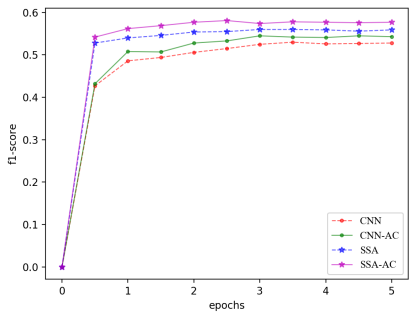


Figure 3: The test curves of f1-score metric with and without AC module w.r.t. the number of training epochs. We take CNN, CNN-AC and SSA, SSA-AC as example.

Table 3: Performance on manually labeled test set. P, R, F1 and GAUC denote micro-precision, micro-recall, micro-F1 and group auc, respectively. The symbol "+" indicates that the higher the value is, the better the model performs. KNN-2 denotes KNN with nearest two neighbors, and so on. DAN-AC denotes DAN base model equipped with our across-context module, and so on. GAUC metric of SGM and SGM-AC are not reported, because only binary relevance information can be obtained from SGM models, without relevant scores.

Models	P(+)	R(+)	F1(+)	GAUC(+)
Click	.798	.147	.248	.566
KNN-2	.780	.301	.435	.624
KNN-5	.751	.365	.491	.631
KNN-10	.727	.406	.521	.631
SGM	.807	.357	.495	-
SGM-AC	.879(**+7.2%**)	.350(-0.7%)	.501(+0.6%)	-
DAN	.871	.415	.563	.780
DAN-AC	.870(-0.1%)	.417(+0.2%)	.564(+0.1%)	.789(**+0.9%**)
CNN	.883	.382	.533	.774
CNN-AC	.884(+0.1%)	.395(**+1.3%**)	.546(**+1.3%**)	.786(**+1.2%**)
BILSTM	.887	.381	.533	.773
BILSTM-AC	.884(-0.3%)	.403(**+2.2%**)	.554(**+2.1%**)	.781(**+0.8%**)
SSA	.876	.411	.560	.797
SSA-AC	.878(+0.2%)	.425(**+1.4%**)	.572(**+1.2%**)	.803(+0.6%)

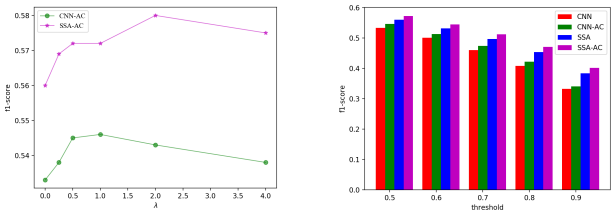
Figure 3 takes CNN, CNN-AC and SSA, SSA-AC as example and presents their f1-score metric on test set with respect to different training epochs. CNN-AC sustainably improves over CNN by a margin, and the same for SSA-AC compared to SSA.

4.6 Analysis and Discussion

In this subsection, we further analysis the impact of hyper-parameters λ, the impact of threshold for final prediction.

4.6.1 Impact of λ. Generally, the performance of our models varies with the coefficient λ that balance direct learning from labeled categories and contrastive learning from query variations. We investigate how the coefficient influences the performance of our models. Taking CNN-AC and SSA-AC as example, we change λ from 0 to 4, and the results are illustrated in Figure 4. Note that $\lambda = 0$ is equivalent to only use base models. The results indicate that across-context module improves base models with different λ. And balance is important to achieve better results. One can set $\lambda = 1.0$ as an initial try.

4.6.2 Impact of threshold. We investigate how the threshold of final score affects model performance. Since our final linear transformation layer has sigmoid activation, 0.5 is a default threshold to decide the relevance of query and category. Taking CNN, CNN-AC and SSA, SSA-AC as example, we increase threshold from 0.5 to 0.9, the results are illustrated in Figure 4. As the increase of threshold, precision metric increase slowly, and recall metric decrease faster, thus f1-score metric decrease for all models. Still, CNN-AC sustainably improves over CNN by a margin, and the same for SSA-AC compared to SSA.



(a) F1-score metric changes with λ coefficient. When $\lambda = 0.0$, it is equivalent to base models. The purple line represents SSA-AC, and green line represents CNN-AC. From left to right, λ increases from 0.0 to 4.0.

(b) F1-score metric decreases as the threshold increases. At each threshold value, we shows f1-score of CNN, CNN-AC, SSA, SSA-AC from left to right. From left to right, threshold increases from 0.5 to 0.9.

Figure 4: Analysis on the impact of λ coefficient and threshold of final score. We take CNN, CNN-AC, SSA, SSA-AC as example.

4.6.3 Impact on different groups of categories. According to the predefined product category taxonomy, we have high-level groups of categories. In our test set, we map each leaf category to its high-level group, and investigate models' performance on different groups of categories with/without across-context component. Taking CNN, CNN-AC, SSA and SSA-AC as example, table 4 shows the precision and recall metrics on category groups with more than one thousand test instances. Our across-context module helps CNN and SSA to achieve better performance on most high-level category groups.

4.7 Case Study

We show several prediction results to demonstrate the properties of across-context module. Our module improves base model SGM in terms of precision metric, and base models like SSA in terms of recall metric.

In table 5, prediction results of SGM and SGM-AC are compared. From these cases we can tell that our across-context module is capable of revising the incorrect category sequences into correct ones. Indeed, SGM may sometimes pay too much attention on the fluency of label sequence, and tend to predict close but incorrect categories. For query "child calcium", the prediction results of SGM are all about health products, while "vitamin" is not "calcium". For query "swimming circle super large", the prediction results of SGM focus on swimming equipment. SGM generates wrong categories like "float board" and "air bed", which are closely related to "swimming circle".

In table 6, prediction results of SSA and SSA-AC are compared. For query "baby small bookshelf simple", SSA-AC is capable to predict the category "residential furniture » bookshelf", which can also be used by baby. For query "inner increase plus velvet shoes men", SSA-AC covers categories "men's shoes»canvas shoes" and "men's shoes»boots" which are missed by SSA.

To further illustrate how the across-context attention module help to understand the original query, we firstly give examples of across-context attention weights in training data by SSA-AC in table 7, and then show related examples of tail queries outside the training data in table 8.

Table 4: Performance on different groups of categories in manually labeled test set. Only groups with more than one thousand instances are represented. P, R denote micro-precision, micro-recall respectively. The symbol "+" indicates that the higher the value is, the better the model performs.

Group of Categories	Models	P(+)	R(+)
Clothing,Shoes,Bags	CNN	.930	.447
Clothing,Shoes,Bags	CNN-AC	.932(+0.2%)	.460(**+1.3%**)
Clothing,Shoes,Bags	SSA	.928	.468
Clothing,Shoes,Bags	SSA-AC	.929(+0.1%)	.477(**+0.9%**)
Sports,Outdoors	CNN	.879	.208
Sports,Outdoors	CNN-AC	.869(**-1.0%**)	.224(**+1.6%**)
Sports,Outdoors	SSA	.887	.232
Sports,Outdoors	SSA-AC	.889(+0.2%)	.246(**+1.4%**)
Home	CNN	.797	.325
Home	CNN-AC	.823(**+2.6%**)	.351(**+2.6%**)
Home	SSA	.787	.381
Home	SSA-AC	.805(**+1.8%**)	.399(**+1.8%**)
3C digital	CNN	.856	.471
3C digital	CNN-AC	.869(**+1.3%**)	.473(+0.2%)
3C digital	SSA	.814	.508
3C digital	SSA-AC	.824(**+1.0%**)	.533(**+2.5%**)
Health	CNN	.727	.352
Health	CNN-AC	.750(**+2.3%**)	.380(**+2.8%**)
Health	SSA	.746	.418
Health	SSA-AC	.744(-0.2%)	.484(**+6.6%**)
Foods	CNN	.731	.405
Foods	CNN-AC	.739(**+0.8%**)	.398(**-0.7%**)
Foods	SSA	.743	.431
Foods	SSA-AC	.743(+0.0%)	.460(**+2.9%**)
Beauty	CNN	.741	.355
Beauty	CNN-AC	.750(**+0.9%**)	.372(**+1.7%**)
Beauty	SSA	.753	.422
Beauty	SSA-AC	.770(**+1.7%**)	.440(**+1.8%**)

As shown in table 7, queries of same category intent share similar distributions of across-context attention weights, while queries of different category intents could have different across-context attention distributions, with respect to same/similar variant queries. Therefore, the difference or similarity of across-context attention distributions on same/similar variant queries can model the contrasting or similar relationships between literally similar queries. For example, "Mahjong small table" is literally similar to "Mahjong table" and "small table folding", but "Mahjong small table" shares similar across-context attention distributions with "Mahjong table" on similar variant queries. From this perspective, "Mahjong small table" is more similar to "Mahjong table" than "small table folding". Therefore, our proposed across-context attention module benefits the identification of original query's category intents.

Table 5: Categories predicted by SGM and SGM-AC for two queries. We mark wrong categories in bold. Each query possesses two lines, firstly the result of SGM, followed by the result of SGM-AC. The prediction of SGM-AC is more precise than SGM.

Query	model	generated category
"child calcium"	SGM	OTC»vitamin minerals health products»calcium **baby nourishment»vitamin** baby nourishment»Ca-Fe-Zn
"child calcium"	SGM-AC	OTC»vitamin minerals health products»calcium baby nourishment»Ca-Fe-Zn
"swimming circle super large"	SGM	**swim»float board** **swim»air bed** swim»swimming circle
"swimming circle super large"	SGM-AC	toys»child swimming circle swim»swimming circle

Table 6: Categories predicted by SSA and SSA-AC for two queries. Each query possesses two lines, firstly the result of SSA, followed by the result of SSA-AC. The prediction of SSA-AC covers more related categories than SSA.

Query	model	predicted relevant category
"baby small bookshelf simple"	SSA	children's furniture»storage rack
"baby small bookshelf simple"	SSA-AC	children's furniture»storage rack residential furniture»bookshelf
"inner increase plus velvet shoes men"	SSA	men's shoes»high top shoes men's shoes»low top shoes
"inner increase plus velvet shoes men"	SSA-AC	men's shoes»high top shoes men's shoes»low top shoes men's shoes»canvas shoes men's shoes»boots

Table 7: Examples of across-context attention weights by SSA-AC in training data. Top three lines are queries of "Mahjong Table" category intent, the last three lines are queries of "Table Household" category intent. With similar variant queries, queries of "Mahjong Table" category intent share similar across-context attention weights, while queries of "Table Household" category intent pay more attention to "Mahjong".

Query	Variant query	AC attention weights
"Mahjong table"	"small Mahjong table"	.247,.356,.397
"Mahjong table square"	"Mahjong table square folding"	.362,.316,.173,.148
"Mahjong small table"	"Mahjong small table folding"	.291,.243,.301,.165
"table"	"Mahjong table"	.681,.319
"table folding"	"Mahjong table folding"	.593,.174,.233
"small table folding"	"Mahjong small table folding"	.515,.033,.227,.225

Table 8 illustrates that tail queries still share similar across-context attention distributions with the frequent queries of same category intents in table 7, under the same variant query "Mahjong small table folding". Thus our across-context attention module is well generalized to tail queries outside the training data. The

Table 8: Examples of across-context attention weights by SSA-AC for tail queries outside the training data. Top four lines are tail queries of "Mahjong Table" category intent, the last four lines are tail queries of "Table Household" category intent. The variant query "Mahjong small table folding" is selected for all queries in this table, to show the across context attention weights, but it does not exist in predicting procedure. Third and fourth columns show the predicted scores of relevant categories by SSA and SSA-AC respectively.

Tail query	AC attention weights	SSA score	SSA-AC score
"simple Mahjong table folding"	.309,.255,.259,.177	.439	.935
"small-sized Mahjong table folding"	.304,.254,.262,.180	.446	.940
"small-sized simple Mahjong table folding"	.327,.235,.234,.205	.399	.776
"mini simple Mahjong table folding"	.318,.248,.249,.184	.004	.940
"simple table folding"	.482,.036,.229,.253	.999	.100
"small-sized table folding"	.519,.034,.218,.230	.100	.100
"small-sized simple table folding"	.492,.034,.230,.244	.970	.100
"mini simple table folding"	.461,.033,.251,.254	.723	.998

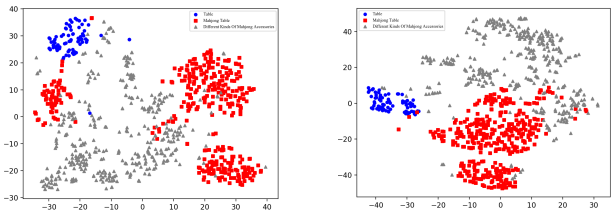
across-context attention distributions indicate that tail query "mini simple Mahjong table folding" is more similar to frequent query "Mahjong small table" than frequent query "small table folding" of "Table Household" category intent, although they are literally similar. Thus, tail query "mini simple Mahjong table folding" may have "Mahjong Table" category intent. The fourth column in table 8 represents the prediction scores of SSA-AC for tail queries and their relevant categories. SSA-AC predicts a score of 0.940 for tail query "mini simple Mahjong table folding" and "Mahjong Table" category, while SSA without across-context module only predicts a score of 0.004. It explains why SSA-AC can recall more relevant categories than SSA.

4.8 Query Representation Visualization

Previous experiments have explored the performance of across-context module quantitatively in detail. To qualitatively study whether across-context module help base model better understand the original query, we dive into the query representation **h** produced by query encoder of base model.

We select two categories as the main objects of study, namely "Table Household" category and "Mahjong Table" category. Some typical queries of "Table Household" category includes: "Nordic dining table","Small table folding" and "Desk mini". And typical queries of "Mahjong Table" can be "Mahjong automatic folding table", "Mahjong table mini" and "Mahjong table dining table dual-use". To better illustrate the difference of query representations, we add queries of Mahjong accessories that belong to several different leaf categories mutually exclusive to "Table Household" category and "Mahjong Table" category. All queries are selected from training data, and are manually labeled as one of the three types.

Figure 5 illustrates the results of base model SSA in the left and our model SSA-AC in the right using t-SNE [24]. In the left, queries of "Mahjong Table" category gather to multiple areas, with queries of Mahjong accessories swarming in. As a contrast, in the right, the result of SSA-AC shows the cohesion of both queries belonging to "Table Household" category and queries belonging to "Mahjong Table" category.



(a) result of SSA (b) result of SSA-AC

Figure 5: Query representations belong to three different types produced by SSA (left) and SSA-AC (right) using t-SNE. Blue circle represents queries of "Table Household" category, red block represents queries of "Mahjong Table" category, and gray triangle represents all kinds of Mahjong accessories that belong to several different leaf categories mutually exclusive to "Table Household" category and "Mahjong Table" category.

Table 9: Online Improvements

Models	PV	CLICK	CVR	PAY
SSA	-	-	-	-
SSA-AC	**+2.02%**	**+0.79%**	**+3.76%**	**+4.59%**

4.9 Online Evaluation

We conduct online evaluation in a real-world e-commerce search engine of *www.taobao.com* with standard A/B testing configuration for one week. We conduct two groups of experiments: predicting categories of infrequent queries by SSA and SSA-AC seperately. For each test, 3% of the users are randomly selected as testing group. The effectiveness is evaluated by some business metrics: Page Views (PV), Item Clicks (CLICK), Conversion Rate (CVR) and Item Pays (PAY). Metrics are calculated under affected query flows.

The result shown in table 9 suggests that by improving the recall of relevant categories, customers tend to view more items. However, since some unpopular items are presented, click through rate can reduce. Due to the increase of page views, more items are clicked. With more choice of items, conversion rate increases dramatically. Finally, more items (+4.59%) are purchased by customers.

5 CONCLUSIONS

In this paper, we propose an across-context attention module for tail query classification in E-commerce. The empirical comparison and online evaluations verify its effectiveness. Beyond product query classification, across-context module can be generalized to other situations where across-context label correlation exists. According to a typical E-commerce knowledge base, there are three types of node: Category, Property Key and Property Value. Our method can be adapted to query property intent prediction, which takes property values as labels. We leave this direction to future work.

REFERENCES

[1] Naomi S Altman. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46, 3 (1992), 175–185.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

[3] Fernando Benites and Elena Sapozhnikova. 2015. Haram: a hierarchical aram neural network for large-scale text classification. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. IEEE, 847–854.

[4] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown. 2004. Learning multi-label scene classification. *Pattern recognition* 37, 9 (2004), 1757–1771.

[5] Daniel Matthew Cer, Yinfei Yang, Sheng yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *ArXiv* abs/1803.11175 (2018).

[6] Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S Dhillon. 2020. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3163–3171.

[7] Guibin Chen, Deheng Ye, Zhenchang Xing, Jieshan Chen, and Erik Cambria. 2017. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. In *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2377–2383.

[8] Weiwei Cheng, Eyke Hüllermeier, and Krzysztof J Dembczynski. 2010. Bayes optimal multilabel classification via probabilistic classifier chains. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 279–286.

[9] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, Jul (2011), 2121–2159.

[10] André Elisseeff and Jason Weston. 2002. A kernel method for multi-labelled classification. In *Advances in neural information processing systems*. 681–687.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[12] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep Unordered Composition Rivals Syntactic Methods for Text Classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, 1681–1691. https://doi.org/10.3115/v1/P15-1162

[13] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972).

[14] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *EACL*.

[15] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1746–1751.

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361, 10 (1995), 1995.

[18] Li Li, Houfeng Wang, Xu Sun, Baobao Chang, Shi Zhao, and Lei Sha. 2015. Multi-label text categorization with joint learning predictions-as-features method. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 835–839.

[19] Junyang Lin, Qi Su, Pengcheng Yang, Shuming Ma, and Xu Sun. 2018. Semantic-Unit-Based Dilated Convolution for Multi-Label Text Classification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 4554–4564.

[20] Yiu-Chang Lin, Ankur Datta, and Giuseppe Di Fabbrizio. 2018. E-commerce Product Query Classification Using Implicit User's Feedback from Clicks. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 1955–1959.

[21] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130* (2017).

[22] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu, and Yiming Yang. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 115–124.

[23] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*. 2873–2879.

[24] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[25] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*. 807–814.

[26] Jinseok Nam, Jungi Kim, Eneldo Loza Mencía, Iryna Gurevych, and Johannes Fürnkranz. 2014. Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 437–452.

[27] Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in neural information processing systems*. 5413–5423.

[28] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*. 1310–1318.

[29] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine learning* 85, 3 (2011), 333.

[30] Dou Shen, Ying Li, Xiao Li, and Dengyong Zhou. 2009. Product query classification. In *Proceedings of the 18th ACM conference on Information and knowledge management*. 741–750.

[31] Amanda Spink, Dietmar Wolfram, Major BJ Jansen, and Tefko Saracevic. 2001. Searching the web: The public and their queries. *Journal of the American society for information science and technology* 52, 3 (2001), 226–234.

[32] Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)* 3, 3 (2007), 1–13.

[33] Zhaopeng Tu, Yang Liu, Zhengdong Lu, Xiaohua Liu, and Hang Li. 2017. Context gates for neural machine translation. *Transactions of the Association for Computational Linguistics* 5 (2017), 87–99.

[34] Ruishuang Wang, Zhenwei Li, Jian Cao, Tong Chen, and Lei Wang. 2019. Convolutional Recurrent Neural Networks for Text Classification. *2019 International Joint Conference on Neural Networks (IJCNN)* (2019), 1–6.

[35] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144* (2016).

[36] Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: Sequence Generation Model for Multi-label Classification. In *Proceedings of the 27th International Conference on Computational Linguistics*. 3915–3926.

[37] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *HLT-NAACL*.

[38] Min-Ling Zhang and Zhi-Hua Zhou. 2006. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering* 18, 10 (2006), 1338–1351.

[39] Min-Ling Zhang and Zhi-Hua Zhou. 2007. ML-KNN: A lazy learning approach to multi-label learning. *Pattern recognition* 40, 7 (2007), 2038–2048.

[40] Han Zhu, Junqi Jin, Chang Tan, Fei Pan, Yifan Zeng, Han Li, and Kun Gai. 2017. Optimized Cost per Click in Taobao Display Advertising. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2017).