

# OTSeq2Set: An Optimal Transport Enhanced Sequence-to-Set Model for Extreme Multi-label Text Classification

**Jie Cao**

Polytechnic Institute  
Zhejiang University  
caojie@zju.edu.cn

**Yin Zhang\***

College of Computer Science and Technology  
Zhejiang University  
yinzhang@zju.edu.cn

## Abstract

Extreme multi-label text classification (XMTC) is the task of finding the most relevant subset labels from an extremely large-scale label collection. Recently, some deep learning models have achieved state-of-the-art results in XMTC tasks. These models commonly predict scores for all labels by a fully connected layer as the last layer of the model. However, such models can't predict a relatively complete and variable-length label subset for each document, because they select positive labels relevant to the document by a fixed threshold or take top k labels in descending order of scores. A less popular type of deep learning models called sequence-to-sequence (Seq2Seq) focus on predicting variable-length positive labels in sequence style. However, the labels in XMTC tasks are essentially an unordered set rather than an ordered sequence, the default order of labels restrains Seq2Seq models in training. To address this limitation in Seq2Seq, we propose an autoregressive sequence-to-set model for XMTC tasks named OTSeq2Set. Our model generates predictions in student-forcing scheme and is trained by a loss function based on bipartite matching which enables permutation-invariance. Meanwhile, we use the optimal transport distance as a measurement to force the model to focus on the closest labels in semantic label space. Experiments show that OTSeq2Set outperforms other competitive baselines on 4 benchmark datasets. Especially, on the Wikipedia dataset with 31k labels, it outperforms the state-of-the-art Seq2Seq method by 16.34% in micro-F1 score. The code is available at <https://github.com/caojie54/OTSeq2Set>.

## 1 Introduction

Extreme multi-label text classification (XMTC) is a Natural Language Processing (NLP) task of finding the most relevant subset labels from an extremely large-scale label set. It has a lot of usage scenarios, such as item categorization in e-commerce and

tagging Wikipedia articles. XMTC become more important with the fast growth of big data.

As in many other NLP tasks, deep learning based models also achieve the state-of-the-art performance in XMTC. For example, AttentionXML (You et al., 2019), X-Transformer (Chang et al., 2020) and LightXML (Jiang et al., 2021) have achieved remarkable improvements in evaluating metrics relative to the current state-of-the-art methods. These models are both composed of three parts (Jiang et al., 2021): text representing, label recalling, and label ranking. The first part converts the raw text to text representation vectors, then the label recalling part gives scores for all cluster or tree nodes including portion labels, and finally, the label ranking part predicts scores for all labels in descending order. Notice that, the label recalling and label ranking part both use fully connected layers. Although the fully connected layer based models have excellent performance, there exists a drawback which is these models can't generate a variable-length and relatively complete label set for each document. Because the fully connected layer based models select positive labels relevant to the document by a fixed threshold or take top k labels in descending order of label scores, which depends on human's decision. Another type of deep learning based models is Seq2seq learning based methods which focus on predicting variable-length positive labels only, such as MLC2Seq (Nam et al., 2017), SGM (Yang et al., 2018). MLC2Seq and SGM enhance Seq2Seq model for Multi-label classification (MLC) tasks by changing label permutations according to the frequency of labels. However, a pre-defined label order can't solve the problem of Seq2Seq based models which is the labels in XMTC tasks are essentially an unordered set rather than an ordered sequence. Yang et al. (2019) solves this problem on MLC tasks via reinforcement learning by designing a reward function to reduce the dependence of the model on the label order, but

\*Corresponding author

it needs to pretrain the model via Maximum Likelihood Estimate (MLE) method. The two-stage training is not efficient for XMTC tasks that have large-scale labels.

To address the above problems, we propose an autoregressive sequence-to-set model, OTSeq2Set, which generates a subset of labels for each document and ignores the order of ground truth in training. OTSeq2Set is based on the Seq2Seq (Bahdanau et al., 2015), which consists of an encoder and a decoder with the attention mechanism. The bipartite matching method has been successfully applied in Named entity recognition task (Tan et al., 2021) and keyphrase generation task (Ye et al., 2021) to alleviate the impact of order in targets. Chen et al. (2019) and Li et al. (2020) have successfully applied the optimal transport algorithm to enable sequence-level training for Seq2Seq learning. Both methods can achieve optimal matching between two sequences, but the difference is the former matches two sequences one to one, and the latter gives a matrix containing regularized scores of all connections. We combine the two methods in our model.

Our contributions of this paper are summarized as follows: (1) We propose two schemes to use the bipartite matching in XMTC tasks, which are suitable for datasets with different label distributions. (2) We combine the bipartite matching and the optimal transport distance to compute the overall training loss, with the student-forcing scheme when generating predictions in the training stage. Our model can avoid the exposure bias; besides, the optimal transport distance as a measurement forces the model to focus on the closest labels in semantic label space. (3) We add a lightweight convolution module into the Seq2Seq models, which achieves a stable improvement and requires only a few parameters. (4) Experimental results show that our model achieves significant improvements on four benchmark datasets. For example, on the Wikipedia dataset with 31k labels, it outperforms the state-of-the-art method by 16.34% in micro-F1 score, and on Amazon-670K, it outperforms the state-of-the-art model by 14.86% in micro-F1 score.

## 2 Methodology

### 2.1 Overview

Here we define necessary notations and describe the Sequence-to-Set XMTC task. Given a text

sequence  $x$  containing  $l$  words, the task aims to assign a subset  $y^g$  containing  $n$  labels in the total label set  $L$  to  $x$ . Unlike fully connected layer based methods which give scores to all labels, the Seq2Set XMTC task is modeled as finding an optimal positive label sequence  $y^g$  that maximizes the joint probability  $P(\hat{y}|x)$ , which is as follows:

$$P(\hat{y}|x) = \prod_{i=1}^n P(y_{\hat{\rho}(i)}|y_1^g, y_2^g, \dots, y_{i-1}^g, x), \quad (1)$$

where  $y^g$  is the sequence generated by the greedy search,  $y$  is the ground truth sequence with default order,  $\hat{y}$  is the most matched reordered sequence computed by bipartite matching. As described in Eq.(1), we use the student-forcing scheme to avoid *exposure bias* (Ranzato et al., 2016) between the generation stage and the training stage. Furthermore, combining the scheme with bipartite matching enables the model to eliminate the influence of the default order of labels.

### 2.2 Sequence-to-Set Model

Our proposed Seq2Set model is based on the Seq2Seq (Bahdanau et al., 2015) model, and the model consists of an encoder and a set decoder with the attention mechanism and an extra lightweight convolution layer (Wu et al., 2019), which are introduced in detail below.

**Encoder** We implement the encoder by a bidirectional GRU to read the text sequence  $x$  from both directions and compute the hidden states for each word as follows:

$$\vec{h}_i = \overrightarrow{\text{GRU}}(\vec{h}_{i-1}, e(x_i)) \quad (2)$$

$$\overleftarrow{h}_i = \overleftarrow{\text{GRU}}(\overleftarrow{h}_{i+1}, e(x_i)) \quad (3)$$

where  $e(x_i)$  is the embedding of  $x_i$ . The final representation of the  $i$ -th word is  $\mathbf{h}_i = [\vec{h}_i; \overleftarrow{h}_i]$  which is the concatenation of hidden states from both directions.

**Attention with lightweight convolution** After the encoder computes  $\mathbf{h}_i$  for all elements in  $x$ , we compute a context vector  $c_t$  to focus on different portions of the text sequence when the decoder generates the hidden state  $s_t$  at time step  $t$ ,

$$c_t = \sum_{i=1}^l \alpha_{ti} \mathbf{h}_i \quad (4)$$

The attention score  $\alpha_{ti}$  of each representation  $h_i$  is computed by

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{j=1}^l \exp(e_{tj})} \quad (5)$$

$$e_{ti} = \mathbf{v}_a^\top \tanh(\mathbf{W}_a \mathbf{s}_{t-1} + \mathbf{U}_a \mathbf{h}_i) \quad (6)$$

where  $\mathbf{W}_a$ ,  $\mathbf{U}_a$ ,  $\mathbf{v}_a$  are weight parameters. For simplicity, all bias terms are omitted in this paper.

To maximally utilize hidden vectors  $\{\mathbf{h}_i\}_{i=1,\dots,l}$  in the encoder, we use the lightweight convolution layer to compute "label" level hidden vectors  $\{\hat{\mathbf{h}}_i\}_{i=1,\dots,k}$ , then compute another context vector  $\hat{\mathbf{c}}_t$  which uses the same parameters as  $\mathbf{c}_t$ ,

$$\{\hat{\mathbf{h}}_i\}_{i=1,\dots,k} = \text{LightConv}(\{\mathbf{h}_i\}_{i=1,\dots,l}). \quad (7)$$

The *lightweight convolutions* are depth-wise separable (Wu et al., 2019), softmax-normalized and share weights over the channel dimension. Readers can refer to Wu et al. (2019) for more details.

**Decoder** The hidden state  $\mathbf{s}_t$  of decoder at time-step  $t$  is computed as follows:

$$\mathbf{s}_t = \text{GRU}(\mathbf{s}_{t-1}, [e(\mathbf{p}_{t-1}); \mathbf{c}_t; \hat{\mathbf{c}}_t]) \quad (8)$$

where  $e(\mathbf{p}_{t-1})$  is the embedding of the label which has the highest probability under the distribution  $\mathbf{p}_{t-1}$ .  $\mathbf{p}_{t-1}$  is the probability distribution over the total label set  $\mathbf{L}$  at time-step  $t-1$  and is computed by a fully connected layer:

$$\mathbf{p}_t = \text{softmax}(\mathbf{W}_p[\mathbf{s}_t; \mathbf{c}_t; \hat{\mathbf{c}}_t; e(\mathbf{p}_{t-1})]) \quad (9)$$

where  $\mathbf{W}_p \in \mathbb{R}^{V \times z}$  is weight parameters.

The overall label size  $V$  of the total label set  $\mathbf{L}$  is usually huge in XMTC. In order to let the model fit into limited GPU memory, we use the hidden bottleneck layer like XML-CNN (Liu et al., 2017) to replace the fully connected layer. The bottleneck layer is described as follows:

$$\mathbf{p}_t = \text{softmax}(\mathbf{W}_{out} \tanh(\mathbf{W}_{in}[\mathbf{s}_t; \mathbf{c}_t; \hat{\mathbf{c}}_t; e(\mathbf{p}_{t-1})])) \quad (10)$$

where  $\mathbf{W}_{in} \in \mathbb{R}^{b \times z}$ ,  $\mathbf{W}_{out} \in \mathbb{R}^{V \times b}$  are weight parameters, and the hyper-parameter  $b$  is the bottleneck size. The size of the parameters in this part reduces from a vast size  $O(V \times z)$  to a much smaller  $O((V+z) \times b)$ . According to the size of labels for different datasets, we can set different  $b$  to make use of GPU memory.

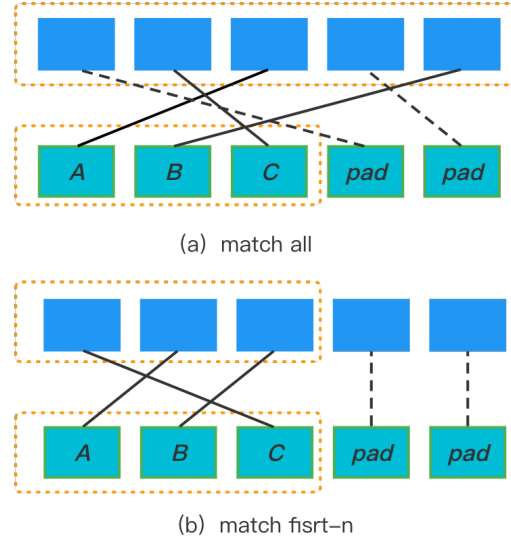


Figure 1: Two assignment schemes.

## 2.3 Loss Function

### 2.3.1 Bipartite Matching

After generating  $N$  predictions in student-forcing scheme, we need to find the most matched re-ordered ground truth sequence  $\hat{\mathbf{y}}$  by bipartite matching between ground truth sequence  $\mathbf{y}$  with default order and the sequence  $\mathbf{p}$  of generated distribution. To find the optimal matching we search for a permutation  $\hat{\rho}$  with the lowest cost:

$$\hat{\rho} = \arg \min_{\rho \in \mathcal{O}_N} \sum_{i=1}^N \mathcal{C}_{\text{match}}(y_{\rho(i)}, \mathbf{p}_i), \quad (11)$$

where  $\mathcal{O}_N$  is the space of all permutations with length  $N$ ,  $\mathcal{C}_{\text{match}}(y_{\rho(i)}, \mathbf{p}_i)$  is a pair matching cost between the ground truth label with index  $\rho(i)$  and generated distribution  $\mathbf{p}_i$  at time step  $i$ . We use the Hungarian method (Kuhn, 1955) to solve this optimal assignment problem. The matching cost considers the generated distributions and is defined as follows:

$$\mathcal{C}_{\text{match}}(y_{\rho(i)}, \mathbf{p}_i) = -\mathbb{1}_{\{y_{\rho(i)} \neq \emptyset\}} \mathbf{p}_i(y_{\rho(i)}), \quad (12)$$

where  $\mathbf{p}_i(y_{\rho(i)})$  is the probability of label  $y_{\rho(i)}$ .  $y_{\rho(i)} \neq \emptyset$  means that the distributions only match with non- $\emptyset$  ground truth labels.

In practice, we design two assignment schemes for bipartite matching. The first scheme is to get the optimal matching between non- $\emptyset$  ground truth labels and all generated distributions, then assign  $\emptyset$  labels to the rest generated distributions one-to-one.

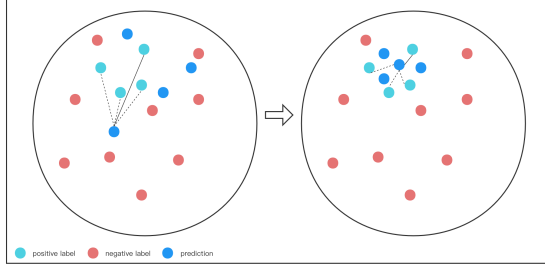


Figure 2: Semantic optimal transport distance in embedding space.

The second scheme is to get the optimal matching between non- $\emptyset$  ground truth labels and first- $n$  generated distributions, then assign  $\emptyset$  labels to the rest generated distributions one-to-one. Figure 1 shows the two assignment schemes.

Finally, we get the bipartite matching loss based on the reordered sequence  $\hat{\mathbf{y}} = \{y_{\hat{\rho}(i)}\}_{i=1, \dots, N}$  to train model, which is defined as follows:

$$\mathcal{L}_b(\hat{\mathbf{y}}, \mathbf{p}) = - \sum_{i=1}^N [\mathbb{1}_{\{y_{\hat{\rho}(i)} \neq \emptyset\}} \log \mathbf{p}_i(y_{\hat{\rho}(i)}) + \mathbb{1}_{\{y_{\hat{\rho}(i)} = \emptyset\}} \lambda_{\emptyset} \log \mathbf{p}_i(y_{\hat{\rho}(i)})] \quad (13)$$

where  $\lambda_{\emptyset}$  is a scale factor less than 1 that forces the model concentrate more on non- $\emptyset$  labels.

### 2.3.2 Semantic Optimal Transport Distance

In XMTC, semantic similar labels commonly appear together in each sample. The bipartite matching loss described in Eq.(13) only utilizes one value in each predictions. We completely utilize the predictions in training, and take the optimal transport (OT) distance in embedding space as a regularization term to make all predictions close to positive labels, as the shown in Figure 2.

**OT distance on discrete domain** The OT distance is also known as Wasserstein distance on a discrete domain  $\mathbb{X}$  (the sequence space), which is defined as follows:

$$\begin{aligned} \mathcal{L}_{ot}(\mathbf{p}, \mathbf{y}) &= \min_{\Gamma \in \Sigma(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \mathbf{C}, \Gamma \rangle \\ &= \min_{\Gamma \in \Sigma(\boldsymbol{\mu}, \boldsymbol{\nu})} \sum_{i=1}^n \sum_{j=1}^m \Gamma_{ij} \cdot c(\mathbf{p}_i, y_j), \end{aligned} \quad (14)$$

where  $\boldsymbol{\mu} \in \mathbb{R}_+^n$ ,  $\boldsymbol{\nu} \in \mathbb{R}_+^m$  are two discrete distributions on  $\mathbb{X}$ ,  $\|\boldsymbol{\mu}\| = \|\boldsymbol{\nu}\| = 1$ . In our case, given realizations  $\{\mathbf{p}_i\}_{i=1}^n$  and  $\{y_j\}_{j=1}^m$  of  $\boldsymbol{\mu}$  and

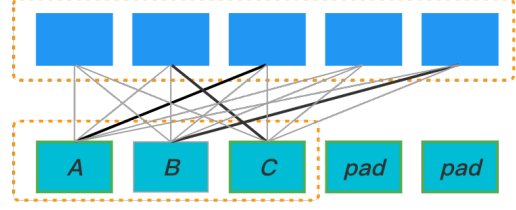


Figure 3: Semantic optimal transport distance.

$\boldsymbol{\nu}$ , we can approximate them by empirical distributions  $\hat{\boldsymbol{\mu}} = \frac{1}{n} \sum \mathbf{p}_i \delta_{\mathbf{p}_i}$  and  $\hat{\boldsymbol{\nu}} = \frac{1}{m} \sum y_j \delta_{y_j}$ . The supports of  $\hat{\boldsymbol{\mu}}$  and  $\hat{\boldsymbol{\nu}}$  are finite, so finally we have  $\boldsymbol{\mu} = \frac{1}{n} \mathbf{1}_{\{\mathbf{p}_i\}}$  and  $\boldsymbol{\nu} = \frac{1}{m} \mathbf{1}_{\{y_j\}}$ .  $\Sigma(\boldsymbol{\mu}, \boldsymbol{\nu})$  is the set of joint distributions whose marginals are  $\boldsymbol{\mu}$  and  $\boldsymbol{\nu}$ , which is described as  $\Sigma(\boldsymbol{\mu}, \boldsymbol{\nu}) = \{\Gamma \in \mathbb{R}_+^{n \times m} | \Gamma \mathbf{1}_m = \boldsymbol{\mu}, \Gamma^\top \mathbf{1}_n = \boldsymbol{\nu}\}$ , where  $\mathbf{1}_n$  represents  $n$ -dimensional vector of ones. Matrix  $\mathbf{C} = [c(\mathbf{p}_i, y_j)] \in \mathbb{R}_+^{n \times m}$  is the cost matrix, whose element  $c(\mathbf{p}_i, y_j)$  denotes the distance between the  $i$ -th support point  $\mathbf{p}_i \in \mathbb{X}$  of  $\boldsymbol{\mu}$  and the  $j$ -th support point  $y_j \in \mathbb{X}$  of  $\boldsymbol{\nu}$ . Notation  $\langle \cdot, \cdot \rangle$  represents the Frobenius dot-product. With the optimal solution  $\Gamma^*$ ,  $\mathcal{L}_{ot}(\mathbf{p}, \mathbf{y})$  is the minimum cost that transport from distribution  $\boldsymbol{\mu}$  to  $\boldsymbol{\nu}$ .

We use a robust and efficient iterative algorithm to compute the OT distance, which is the recently introduced Inexact Proximal point method for exact Optimal Transport (IPOT) (Xie et al., 2020).

**Semantic Cost Function in OT** In XMTC learning, it's intractable to directly compute the cost distance between predictions and ground-truth one-hot vector because of the huge label space. A more natural choice for computing cost distance is to use the label embeddings. Based on the rich semantics of embedding space, we can compute the semantic OT distance between model predictions and ground-truth. The details of computing the *cosine distance* are described as follows:

$$c(\mathbf{p}_i, y_j) = \text{cosine\_distance}(\mathbf{E}^\top \mathbf{p}_i, \mathbf{E}^\top \mathbb{1}_{y_j}) \quad (15)$$

$$\text{cosine\_distance}(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^\top \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} \quad (16)$$

where  $\mathbf{E} \in \mathbb{R}^{V \times d}$  is the label embedding matrix,  $V$  is the vocabulary size and  $d$  is the dimension for the embedding vector.  $\mathbb{1}_{y_j}$  is the one-hot vector whose value is 0 in all positions except at position  $y_j$  is 1.

In practice, we compute the semantic optimal transport distance  $\mathcal{L}_{ot}(\mathbf{p}, \mathbf{y})$  between non- $\emptyset$



ground-truth labels and all predictions, which is shown in 3.

### 2.3.3 Complete Seq2Set Training with OT regularization

The bipartite matching gives the optimal matching between ground truth and predictions, which enables the Seq2Set model to train with Maximum likelihood estimation (MLE). In addition, the OT distance provides a measurement in semantic label embedding space. We propose to combine the OT distance and the bipartite matching loss, which gives us the final training objective:

$$\mathcal{L} = \mathcal{L}_b(\hat{\mathbf{y}}, \mathbf{p}) + \lambda \mathcal{L}_{ot}(\mathbf{p}, \mathbf{y}) \quad (17)$$

where  $\lambda$  is a hyper-parameter to be tuned.

## 3 Experiments

### 3.1 Datasets

We use four benchmark datasets, including three large-scale datasets Eurlex-4K (Loza Mencía and Fürnkranz, 2008) and AmazonCat-13K (McAuley and Leskovec, 2013) and Wiki10-31K (Zubiaga, 2012), one extreme-scale dataset Amazon-670K (McAuley and Leskovec, 2013). We follow the training and test split of AttentionXML (You et al., 2019) and we set aside 10% of the training instances as the validation set for hyperparameter tuning. The dataset statistics are summarized in Table 1.

### 3.2 Evaluation Metrics

Following the previous work (Yang et al., 2018, Nam et al., 2017, Yang et al., 2019), we evaluate all methods in terms of micro-F1 score, micro-precision and micro-recall are also reported for analysis.

### 3.3 Baselines

We compare our proposed methods with the following competitive baselines:

**Seq2Seq** (Bahdanau et al., 2015) is a classic Seq2Seq model with attention mechanism, which is also a strong baseline for XMTC tasks.

**MLC2Seq** (Nam et al., 2017) is based on Seq2Seq (Bahdanau et al., 2015), which enhances Seq2Seq model for MLC tasks by changing label permutations according to the frequency of labels. We take the descending order from frequent to rare in our experiments.

**SGM** (Yang et al., 2018) is based on previews Seq2Seq (Bahdanau et al., 2015) and MLC2Seq (Nam et al., 2017), which views the MLC task as a sequence generation problem to take the correlations between labels into account.

In order to extend the above three models to large-scale dataset Wiki10-31K and extreme-scale dataset Amazon-670K, we use the bottleneck layer to replace the fully connected layer. The bottleneck sizes are same as our proposed model and they are shown in Table 2.

### 3.4 Experiment Settings

For each dataset, the vocabulary size of input text is limited to 500,000 words according to the word frequency in the dataset.

The number of GRU layers of the encoder is 2, and for the decoder is 1. The hidden sizes of the encoder and the decoder both are 512. We set the max length of generated predictions  $N$  to the max length of labels in each dataset are shown in Table 1 as  $L_{max}$ .

For LightConv, we set the kernel sizes to 3,7,15,30 for each layer respectively. To reduce the memory consumption of GPU, we set the stride of convolution to 3 in the last layer for all datasets except that Wiki10-31K is 4.

We set the hyperparameter  $\lambda_{\emptyset}$  of bipartite matching loss to 0.2 in Eq.(13) following Ye et al. (2021), and  $\lambda$  of final loss to 8 in Eq.(17).

For word embeddings in three baseline models and our proposed models, we use pre-trained 300-dimensional Glove vectors (Pennington et al., 2014) to all datasets for input text, and use the mean value of embeddings of words in labels for all datasets except for Amazon-670K, because the labels in Amazon-670K are corresponding item numbers and we randomly initialize 100-dimensional embeddings for Amazon-670K.

All models are trained by the Adam optimizer (Kingma and Ba, 2014) with a cosine annealing schedule. Besides, we use dropout (Srivastava et al., 2014) to avoid overfitting, the dropout rate is 0.2, and clip the gradients (Pascanu et al., 2013) to the maximum norm of 8.

All models are trained on one Nvidia TITAN V and one Nvidia GeForce RTX 3090, for the small dataset Eurlex-4K, we use TITAN V, and for others, we use RTX 3090.

Other hyperparameters are given in Table 2.

Dataset	$N_{train}$	$N_{test}$	$D$	$L$	$\bar{L}$	$L_{max}$	$\hat{L}$	$\bar{W}_{train}$	$\bar{W}_{test}$
Eurlex-4K	15,449	3,865	186,104	3,956	5.30	24	20.79	1248.58	1230.40
Wiki10-31K	14,146	6,616	101,938	30,938	18.64	30	8.52	2484.30	2425.45
AmazonCat-13K	1,186,239	306,782	203,882	13,330	5.04	57	448.57	246.61	245.98
Amazon-670K	490,449	153,025	135,909	670,091	5.45	7	3.99	247.33	241.22

Table 1: Data Statistics.  $N_{train}$  and  $N_{test}$  refer to the number of training samples and test samples respectively.  $D$  refers to the total number of features.  $L$  is the total number of labels,  $\bar{L}$  is the average number of labels per training sample,  $L_{max}$  is the max length of labels for both training and test samples,  $\hat{L}$  is the average number of samples per training label.  $\bar{W}_{train}$  and  $\bar{W}_{test}$  refer to the average number of words per training and test sample respectively.

Datasets	$E$	$B$	$lr$	$b$	$L_t$
Eurlex-4K	20	32	0.001	-	1000
Wiki-31K	15	32	0.001	300	2000
AmazonCat-13K	5	64	0.0005	-	500
Amazon-670K	20	64	0.0005	512	500

Table 2: Hyperparameters of all datasets.  $E$  is the number of epochs,  $B$  is the batch size,  $lr$  is init learning rate,  $b$  is the bottleneck size (Note that the bottleneck layer is not used on Eurlex-4K and AmazonCat-13K),  $L_t$  is the maximum length of input tokens.

### 3.5 Main Results

Table 3 compares the proposed methods with three baselines on four benchmark datasets. We focus on micro-F1 score following previous works based on Seq2Seq. The best score of each metric is in boldface. Our models outperform all baselines in micro-F1 score. We find that the two assignment scheme of bipartite matching each excel on different datasets.  $BM_{all}$  denotes the Seq2Seq model with the first scheme of bipartite matching,  $BM_{first-n}$  denotes the Seq2Seq model with the second one.  $BM_{all}$  has the better performance on Eurlex-4K and Amazon-670K, rather  $BM_{first-n}$  is better on Wiki10-31K and AmazonCat-13K. Our complete method achieves a large improvement of 16.34% micro-F1 score over the second best baseline model which is SGM on Wiki10-31K, and 14.86%, 7.26%, 1.01% on other datasets respectively, the relative improvements are shown in Table 4.

### 3.6 $BM_{first-n}$ vs $BM_{all}$

We find that the different performance of  $BM_{first-n}$  and  $BM_{all}$  are related to the distribution of label size on the dataset. Table 5 compares  $BM_{first-n}$  with  $BM_{all}$  in micro-F1 score on all datasets. The performance difference on Eurlex-4K is very small since the two proportions which are shown in Figure 4 are nearly equal. Wiki10-31K is

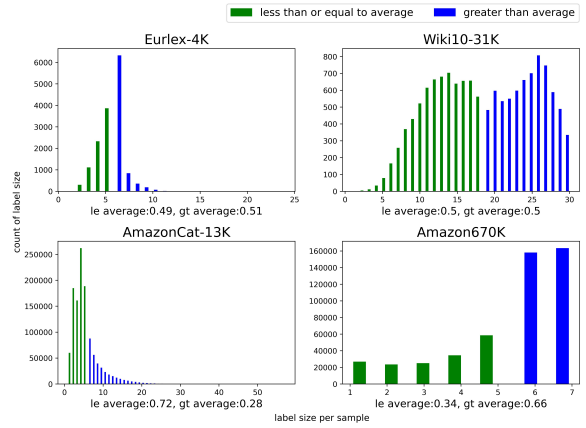


Figure 4: Distributions of label size on all datasets. The  $le$  average denotes the proportion of samples whose label size less than or equal to average number of label size, the  $gt$  average denotes the opposite proportion.

the same case. However, the two proportions have a big difference on AmazonCat-13K and Amazon-670K, which leads to the large performance difference between  $BM_{first-n}$  and  $BM_{all}$ . The proportion of samples whose label size is smaller than the average is less than 50%, then the performance of  $BM_{first-n}$  is better than  $BM_{all}$ . The proportion of samples whose label size is greater than average number is greater than 50%, then the performance of  $BM_{all}$  is better than  $BM_{first-n}$ .

### 3.7 Effect of Lightweight Convolution and Semantic Optimal Transport Distance

To examine the effect of LightConv and OT, we add LightConv to the Seq2seq model and BM model, and add OT to BM model. The details of results are shown in Table 3. The semantic optimal transport distance and LightConv achieve improvements against origin models on all datasets except for Amazon-670K, but it's not beyond our imagination, because Amazon-670K has an extremely large label set which is item numbers. The item numbers don't have semantic information leading to OT hav-

Methods	P	R	F1	Methods	P	R	F1
Eurlex-4K				Wiki10-31K			
Seq2Seq	60.83	54.76	57.64	Seq2Seq	29.87	26.96	28.34
MLC2Seq	59.80	56.70	58.21	MLC2Seq	28.69	28.03	28.36
SGM	59.99	56.56	58.23	SGM	28.89	27.93	28.40
Seq2Seq + LC	61.36	55.03	58.02	Seq2Seq + LC	30.06	27.14	28.53
BM <sub>all</sub>	61.72	57.63	59.61	BM <sub>first-n</sub>	33.23	29.43	31.22
BM <sub>all</sub> + LC	63.38	58.10	60.63	BM <sub>first-n</sub> + LC	33.58	29.94	31.66
BM <sub>all</sub> + OT	<b>65.95</b>	58.54	62.03	BM <sub>first-n</sub> + OT	<b>37.26</b>	29.08	32.67
BM <sub>all</sub> + OT + LC	64.82	<b>60.26</b>	<b>62.46</b>	BM <sub>first-n</sub> + OT + LC	36.70	<b>30.05</b>	<b>33.04</b>
Amazon-670K				AmazonCat-13K			
Seq2Seq	32.16	32.04	32.10	Seq2Seq	74.26	68.24	71.12
MLC2Seq	30.27	<b>32.32</b>	31.26	MLC2Seq	70.93	69.66	70.29
SGM	30.12	31.42	30.76	SGM	71.27	69.36	70.30
Seq2Seq + LC	31.15	31.32	31.24	Seq2Seq + LC	<b>74.60</b>	68.49	71.41
BM <sub>all</sub>	55.23	27.33	36.56	BM <sub>first-n</sub>	70.26	72.51	71.37
BM <sub>all</sub> + LC	52.70	26.30	35.09	BM <sub>first-n</sub> + LC	70.46	<b>72.95</b>	71.68
BM <sub>all</sub> + OT	58.36	26.46	36.42	BM <sub>first-n</sub> + OT	71.80	71.57	71.69
BM <sub>all</sub> + OT + LC	<b>61.16</b>	26.39	<b>36.87</b>	BM <sub>first-n</sub> + OT + LC	71.98	71.71	<b>71.84</b>

Table 3: Comparison between our models and three baselines on Eurlex-4K, Wiki10-31K, Amazon-670K, and AmazonCat-13K. BM<sub>all</sub> denotes the Seq2Seq model with the first scheme of bipartite matching loss, BM<sub>first-n</sub> denotes the Seq2Seq model with the second one. OT denotes the semantic optimal transport distance loss. LC denotes the light weight convolution layer. We take the average score of 4 times experiments for all models on Eurlex-4K and Wiki10-31K, and 2 times experiments for the other two datasets.

Methods	Eurlex-4K	Wiki10-31K	Amazon-670K	AmazonCat-13K
BM	2.37%	9.93%	13.89%	0.35%
BM + OT	6.53%	15.04%	13.46%	0.8%
BM + OT + LC	7.26%	16.34%	14.86%	1.01%

Table 4: Relative improvements of micro-F1 score over the second best baseline model on four datasets in Table 3. BM denotes BM<sub>all</sub> or BM<sub>first-n</sub>.

Datasets	BM <sub>all</sub>	BM <sub>first-n</sub>
Eurlex-4K	59.61	59.12(↓ 0.82%)
Wiki10-31K	31.38	31.22(↓ 0.51%)
AmazonCat-13K	67.09	71.37(↑ 6.38%)
Amazon-670K	36.87	34.45(↓ 6.56%)

Table 5: The micro-F1 score of BM<sub>all</sub> and BM<sub>first-n</sub> and relative difference on four datasets.

ing no effect. At the same time Amazon-670K has the lowest average number of samples per label, it’s very hard to learn a proper context vector by the LightConv. However, the combination of OT and LightConv still has a small improvement over BM<sub>all</sub> methods on Amazon-670K.

### 3.8 Comprehensive Comparison

To realize the performance of our method on tail labels, we use the macro-averaged F1 (maF1) score which treats all labels equally regardless of their support values. To more comprehensively compare our model with baselines, we also use the weighted-averaged F1 (weF1) which considers each label’s support, and the example-based F1 (ebF1) which calculates F1 score for each instance and finds their average. We show the results in Table 6.

### 3.9 Performance over different $\lambda$

We conduct experiments on Eurlex4K dataset to evaluate performance with different hyperparameter  $\lambda$  in complete loss function  $\mathcal{L}$ . The results are shown in Figure 5. The figure shows that when  $\lambda = 8$ , the performance reaches its best and is stable.

Methods	maF1	weF1	ebF1	Methods	maF1	weF1	ebF1
Eurlex-4K				Wiki10-31K			
Seq2Seq	23.26	55.36	57.10	Seq2Seq	3.51	24.55	27.97
MLC2Seq	24.16	56.24	58.00	MLC2Seq	<b>4.85</b>	26.15	28.14
SGM	24.32	56.28	57.96	SGM	4.29	26.00	28.19
OTSeq2Set <sub>all</sub>	<b>27.34</b>	<b>60.45</b>	<b>61.04</b>	OTSeq2Set <sub>first-n</sub>	3.50	<b>26.55</b>	<b>32.65</b>
Amazon-670K				AmazonCat-13K			
Seq2Seq	<b>11.57</b>	30.29	<b>29.62</b>	Seq2Seq	41.28	69.37	74.03
MLC2Seq	11.55	<b>30.36</b>	<b>29.62</b>	MLC2Seq	39.79	68.78	73.57
SGM	11.14	29.53	28.85	SGM	40.44	68.96	73.68
OTSeq2Set <sub>all</sub>	9.91	28.83	25.65	OTSeq2Set <sub>first-n</sub>	<b>43.42</b>	<b>70.95</b>	<b>75.18</b>

Table 6: Performance comparison of all models on the macro-averaged F1, the weighted- averaged F1 and the example-based F1.

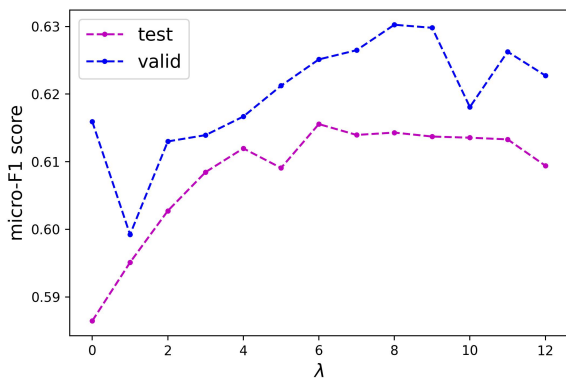


Figure 5: Performance of the OTSeq2Set using different  $\lambda$  in loss function.

### 3.10 Computation Time and Model Size

Table 7 shows the overall training time of all models compared in our experiments. The MLC2Seq has the same model size and training time as Seq2Seq. The model size of OTSeq2Set is almost the same as that of Seq2Seq, but the overall training time is much longer than other models, because the bipartite matching loss and the optimal transport distance need to compute for each sample individually.

## 4 Related Work

The most popular deep learning models in XMTC are fully connected layer based models. XML-CNN (Liu et al., 2017) uses a convolutional neural network (CNN) and a dynamic max pooling scheme to learn the text representation, and adds an hidden bottleneck layer to reduce model size as well as boost model performance. AttentionXML (You et al., 2019) utilizes a probabilistic label tree (PLT) to handle millions of labels. In At-

Datasets		Seq2Seq	SGM	OTSeq2Set
Eurlex-4K	$T_{train}$	0.85	0.85	1.52
	$M$	34.8	35.0	40.5
Wiki10-31K	$T_{train}$	1.55	1.56	1.78
	$M$	45.5	45.7	47.5
AmazonCat-13K	$T_{train}$	3.70	3.75	17.88
	$M$	54.8	55.0	70.1
Amazon-670K	$T_{train}$	12.9	16.0	18.5
	$M$	437.7	437.7	439.8

Table 7: Training Time and Model Size.  $T_{train}$  is the overall training hours.  $M$  is the total size of model parameters in millions.

tentionXML, a BiLSTM is used to capture long-distance dependency among words and a multi-label attention is used to capture the most relevant parts of texts to each label. CorNet (Xun et al., 2020) proposes a correlation network (CorNet) capable of utilizing label correlations. X-Transformers (Chang et al., 2020) proposes a scalable approach to fine-tuning deep transformer models for XMTC tasks, which is also the first method using deep transformer models in XMTC. LightXML (Jiang et al., 2021) combines the transformer model with generative cooperative networks to fine-tune transformer model.

Another type of deep learning-based method is Seq2Seq learning based methods. MLC2Seq (Nam et al., 2017) is based on the classic Seq2Seq (Bahdanau et al., 2015) architecture which uses a bidirectional RNN to encode raw text and an RNN with an attention mechanism to generate predictions sequentially. MLC2Seq enhances the model performance by determining label permutations before training. SGM (Yang et al., 2018) proposes a



novel decoder structure to capture the correlations between labels. Yang et al. (2019) proposes a two-stage training method, which is trained with MLE then is trained with a self-critical policy gradient training algorithm.

About bipartite matching, Tan et al. (2021) considers named entity recognition as a Seq2Set task, the model generates an entity set by a non-autoregressive decoder and is trained by a loss function based on bipartite matching. ONE2SET (Ye et al., 2021) proposes a K-step target assignment mechanism via bipartite matching on the task of keyphrases generation, which also uses a non-autoregressive decoder.

Xie et al. (2020) proposes a fast proximal point method to compute the optimal transport distance, which is named as IPOT. Chen et al. (2019) adds the OT distance as a regularization term to the MLE training loss. The OT distance aims to find an optimal matching of similar words/phrases between two sequences which promotes their semantic similarity. Li et al. (2020) introduce a method that combines the student-forcing scheme with the OT distance in text generation tasks, this method can alleviate the exposure bias in Seq2Seq learning. The above two methods both use the IPOT to compute the OT distance.

## 5 Conclusion

In this paper, we propose an autoregressive Seq2Set model for XMTC, OTSeq2Set, which combines the bipartite matching and the optimal transport distance to compute overall training loss and uses the student-forcing scheme in the training state. OTSeq2Set not only eliminates the influence of order in labels, but also avoids the exposure bias. Besides, we design two schemes for the bipartite matching which are suitable for datasets with different label distributions. The semantic optimal transport distance can enhance the performance by the semantic similarity of labels. To take full advantage of the raw text information, we add a lightweight convolution module which achieves a stable improvement and requires only a few parameters. Experiments show that our method gains significant performance improvements against strong baselines on XMTC.

## Limitations

For better effect, we compute the bipartite matching and the optimal transport distance between non- $\emptyset$  targets and predictions. We can't use batch com-

putation to improve efficiency so that the training time of OTSeq2Set is longer than other baseline models.

## Acknowledgement

This study is supported by China Knowledge Centre for Engineering Sciences and Technology (CKCEST).

## References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.
- Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S. Dhillon. 2020. [Taming pre-trained transformers for extreme multi-label text classification](#). In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20*, page 3163–3171, New York, NY, USA. Association for Computing Machinery.
- Liqun Chen, Yizhe Zhang, Ruiyi Zhang, Chenyang Tao, Zhe Gan, Haichao Zhang, Bai Li, Dinghan Shen, Changyou Chen, and Lawrence Carin. 2019. [Improving sequence-to-sequence learning via optimal transport](#). In *International Conference on Learning Representations*.
- Ting Jiang, Deqing Wang, Leilei Sun, Huayi Yang, Zhengyang Zhao, and Fuzhen Zhuang. 2021. [Lightxml: Transformer with dynamic negative sampling for high-performance extreme multi-label text classification](#). In *AAAI*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.
- Jianqiao Li, Chunyuan Li, Guoyin Wang, Hao Fu, Yuhchen Lin, Liqun Chen, Yizhe Zhang, Chenyang Tao, Ruiyi Zhang, Wenlin Wang, Dinghan Shen, Qian Yang, and Lawrence Carin. 2020. [Improving text generation with student-forcing optimal transport](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9144–9156. Association for Computational Linguistics.
- 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*, pages 115–124. ACM.

- Eneldo Loza Mencía and Johannes Fürnkranz. 2008. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *Machine Learning and Knowledge Discovery in Databases*, pages 50–65. Springer Berlin Heidelberg.
- Julian McAuley and Jure Leskovec. 2013. [Hidden factors and hidden topics: understanding rating dimensions with review text](#). In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM.
- 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*, volume 30. Curran Associates, Inc.
- Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. [Glove: Global vectors for word representation](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.
- Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. *CoRR*, abs/1511.06732.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.
- Zeqi Tan, Yongliang Shen, Shuai Zhang, Weiming Lu, and Yueting Zhuang. 2021. [A sequence-to-set network for nested named entity recognition](#). In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 3936–3942. International Joint Conferences on Artificial Intelligence Organization.
- Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. [Pay less attention with lightweight and dynamic convolutions](#). In *International Conference on Learning Representations*.
- Yujia Xie, Xiangfeng Wang, Ruijia Wang, and Hongyuan Zha. 2020. A fast proximal point method for computing exact wasserstein distance. In *Uncertainty in artificial intelligence*, pages 433–453. PMLR.
- Guangxu Xun, Kishlay Jha, Jianhui Sun, and Aidong Zhang. 2020. [Correlation networks for extreme multi-label text classification](#). In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1074–1082. ACM.
- Pengcheng Yang, Fuli Luo, Shuming Ma, Junyang Lin, and Xu Sun. 2019. [A deep reinforced sequence-to-set model for multi-label classification](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5252–5258. Association for Computational Linguistics.
- 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*, pages 3915–3926. Association for Computational Linguistics.
- Jiacheng Ye, Tao Gui, Yichao Luo, Yige Xu, and Qi Zhang. 2021. [One2set: Generating diverse keyphrases as a set](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4598–4608. Association for Computational Linguistics.
- Ronghui You, Zihan Zhang, Ziyue Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. Attentionxml: Label tree-based attention-aware deep model for high-performance extreme multi-label text classification. In *NeurIPS*.
- Arkaitz Zubiaga. 2012. [Enhancing navigation on wikipedia with social tags](#). Number: arXiv:1202.5469.