Cluster-Guided Label Generation in Extreme Multi-Label Classification

Taehee Jung ♣ * **Joo-Kyung Kim** ♣ **Sungjin Lee** ♣ **Dongyeop Kang** ♡

♣Amazon Alexa AI ♡University of Minnesota
{jungtaeh,jookyk,sungjinl}@amazon.com dongyeop@umn.edu

Abstract

For extreme multi-label classification (XMC), existing classification-based models poorly perform for tail labels and often ignore the semantic relations among labels, like treating "Wikipedia" and "Wiki" as independent and separate labels. In this paper, we cast XMC as a *generation* task (XLGen), where we benefit from pre-trained text-to-text models. However, generating labels from the extremely large label space is challenging without any constraints or guidance. We, therefore, propose to guide label generation using label cluster information to hierarchically generate lower-level labels. We also find that frequency-based label ordering and using decoding ensemble methods are critical factors for the improvements in XLGen. XLGen with cluster guidance significantly outperforms the classification and generation baselines on tail labels, and also generally improves the overall performance in four popular XMC benchmarks. In human evaluation, we also find XLGen generates unseen but plausible labels. Our code is now available at https://github.com/alexa/xlgen-eacl-2023.

1 Introduction

Extreme multi-label classification (XMC) is a task to predict multiple relevant labels for a given input where the label space is extremely large. Conventional approaches for XMC decompose the problem into a set of binary classifications, training one-vs-all classifiers for each label. However, they encounter several issues in practical use cases.

First, the labels in XMC are long-tail distributed. In other words, only a few labels have sufficient positive samples, thereby the other infrequent labels could be rarely predicted during inference as we see the heavily right-skewed distribution in the long-tail in Figure 2a. Second, multi-label classification techniques such as one-by-one and label powerset (Gibaja, 2015) assume independent and identically distributed labels, while the user-generated labels in XMC are dependent on each other. Moreover, annotated labels are only a portion of possible labels, thus, resulting in positive and unlabeled (PU) setting (Yu et al., 2014; Kanehira and Harada, 2016).

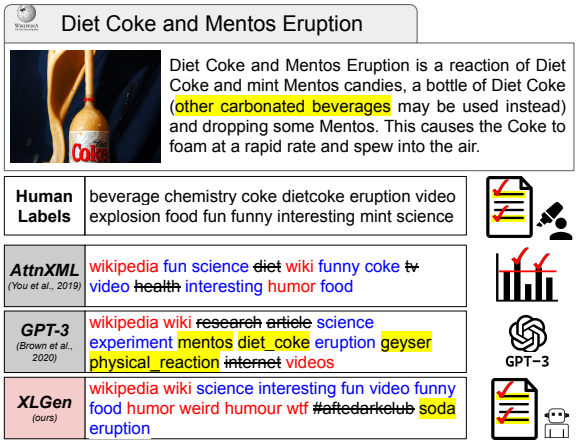
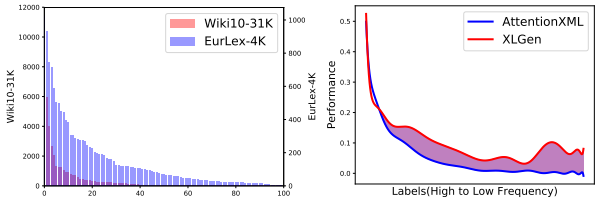


Figure 1: The predicted and generated labels from AttentionXML (You et al., 2019), GPT-3 (Brown et al., 2020), and XLGen-BCL, respectively, for Wikipedia page on diet coke and mentos eruption. We marked labels to be correct (blue), wrong (~~strikethrough~~), and positive unlabeled (red). Our XLGen could generate completely new labels from input text, e.g., soda, inferred from context that other carbonated beverages can replace diet coke.

In this paper, we tackle extreme multi-label classification with a *generative approach*, called extreme multi-label generation (XLGen). In particular, we fine-tune a pre-trained Transformer-based encoder-decoder model (Raffel et al., 2020) with input documents and their known positive labels. This (label) generation approach is more intuitive and closely similar to how humans tag documents with text labels without a fine-grained ontology or guideline.

However, the generated labels from the extremely large label space without any constraints

* Part of this work was done during an internship at Amazon Alexa AI.



(a) Frequency Histogram (b) Label-wise Recall

Figure 2: (a) Frequency histograms of top-100 occurring labels in EUR-LEX (blue) and WIKI10-31K (red) (b) Label-wise recall scores from AttentionXML and XLGen-BCL on Wiki10-31K. For the presentation, graphs are smoothed by least-squared polynomial regression.

and/or guidance can be noisy and not cover infrequent labels. To address this issue, we propose a method to leverage label clusters into generation: first, generate cluster IDs of semantically similar labels, and then generate text labels utilizing the cluster IDs as additional contextual inputs. Specifically, we propose two XLGen architectures (XLGen-BCL, XLGen-MCG) in which such clusters are jointly trained with labels in different ways. Using clusters for label generation is motivated by showing label categories to human annotators. As an example, humans often start by setting high-level topics first and then hierarchically create actual tags under each high-level topic. The clusters, however, are treated as additional guidance rather than a constraint since we do not restrict the model to only predict labels under the given clusters.

Similarly to XLGen, Simig et al. (2022) proposed GROOV, which fine-tunes T5 to generate labels in XMC. In particular, GROOV aims to implement a label order invariant training objective by randomly shuffling label orders and using multi-softmax function, which does not penalize if any first tokens of true labels are predicted regardless of the label orders. However, it does not outperform classification baselines consistently, and we empirically find that label order by frequency helps alleviate the issue in our ablation study.

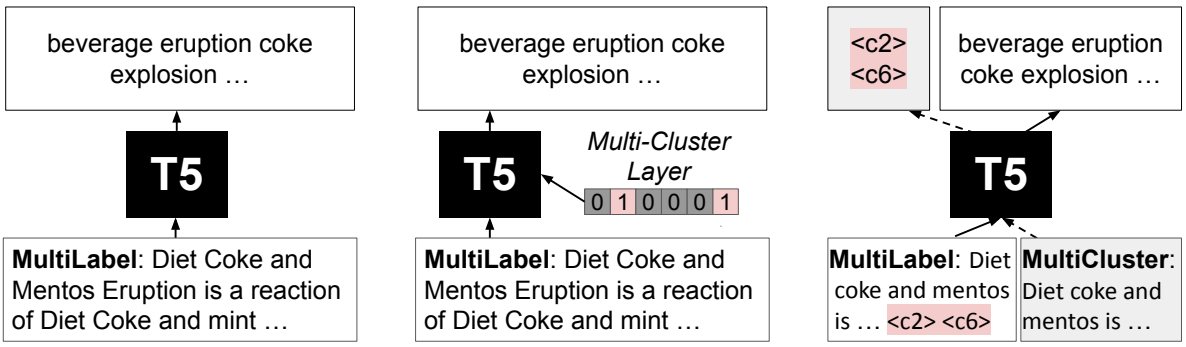
Our experiment shows that XLGen (and its variants) outperforms classification baselines on four XMC benchmarks. Furthermore, XLGen with cluster guidance (XLGen-BCL and -MCG) significantly and consistently outperforms the classification and generation baseline (XLGen-base) on tail labels, respectively. The effect on tail labels from XLGen-BCL is demonstrated in Figure 2b.

Figure 1 shows predicted or generated labels from different models. A Wikipedia page of diet coke and mentos eruption has true labels such as "beverage", "fun", and "eruption". We find XLGen can also generate a new positive label "soda" based on the context of "carbonated beverage" in the input text. From a human evaluation (S6.1), we find newly generated labels by XLGen are highly associated with the input texts, which potentially helps automatically find new labels without manual tagging. We also show the generated labels from large language models (LLMs) like GPT-3 (Brown et al., 2020): we find that the overall performance of in-context learning is significantly less than XLGen (See §4.4 for details), but LLMs could generate reasonable labels with a few examples as XLGen does.

2 Related Work

Extreme multi-label classification (XMC). Classification-based approaches on XMC suffer from dealing with enormous label spaces under the one-vs-all classification setting (Babbar and Schölkopf, 2017; Yen et al., 2017; Jain et al., 2019). To address the efficiency issue, state-of-the-art XMC models partition label space to the scalable subsets via hierarchical clustering (Prabhu et al., 2018; Wydmuch et al., 2018; You et al., 2019; Chang et al., 2020; Yu et al., 2022; Tagami, 2017), graph-based approximations (Jain et al., 2019; Zong and Sun, 2022), or random forest (Siblini et al., 2018). However, they still suffer from predicting tail or unseen labels. To efficiently deal with such long-tail issues, few-shot learning frameworks and methods (Gupta et al., 2021; Xiong et al., 2022) are proposed. Rather, we show how encoder-decoder language model can improve tail label scores by fine-tuning it with guidance from label clusters.

LMs and Generative approach in XMC. For XMC, pre-trained LMs such as XLNet (Ye et al., 2020) and Transformer (Chang et al., 2020) are used but only for encoding input texts, thus, it still relies on the classification approach for label prediction. Previously, other works address multi-label classification with generative approaches (Nam et al., 2017; Tsai and Lee, 2020; Yang et al., 2018, 2019; Zhang et al., 2021b) but in much smaller label spaces. Recently, Simig et al. (2022) also used T5 (Raffel et al., 2020) for directly generating labels in end-to-end manners for XMC, but



(a) XLGen-base (b) XLGen-BCL (c) XLGen-MCG

Figure 3: Three XLGen architectures, where the basic model can be any pre-trained text-to-text models like T5. (a, base): Simple fine-tuning that encodes input text with a prefix of task name and decodes text of label sequences. (b, BCL): A fine-tuning with a multi-cluster prediction layer as an auxiliary task. (c, MCG): A multi-task fine-tuning with multi-cluster generation and multi-label generation (MCG); two tasks are trained simultaneously and at decoding time the output of the cluster generation is concatenated to the input for the label generation.

its performance was not convincing compared to classification-based models.

Positive and unlabeled data. In practice, XMC is inherently with positive and unlabeled (PU) setting as the label space is extremely large and it is infeasible to manually review all the labels (Kim and Kim, 2020). Multi-label performances on PU tasks can be simulated by leaving only a few labels per train instance (e.g., leaving 8 out of 10 positive labels in one instance for label deficit rate 20%) positive (Hu et al., 2021). In this work, we show how XLGen works on such PU settings in §4.3.

3 Extreme Multi-label Generation (XLGen) with Cluster Guidance

Classifying a document with multiple labels can be regarded as tagging a document with possible topical labels, which is basically decoding the free-formed text labels in an encoder-decoder setting. Moreover, if encoder and decoder are trained on large text corpora, labels are generated with an understanding of their lexical variations and semantic similarities. Our baseline framework fine-tunes a pre-trained Transformer using the input text as an encoder input and the label sequences as a decoder output. In addition, to more effectively address a huge number of labels in a long-tail distribution, we propose two different architectures, XLGen-BCL (§3.2) and XLGen-MCG (§3.3), generating labels guided by pre-computed cluster information, inspired by class-based language models (Brown et al., 1992).

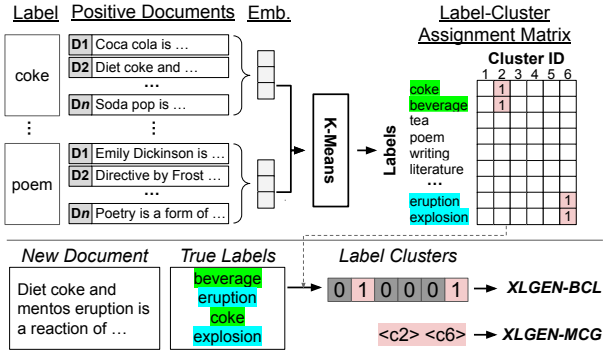


Figure 4: Architecture of pre-computed label clusters. For each label, we use the averaged embedding of positive documents including the label in train set and compute the label cluster assignment matrix (top). For training, we assign label clusters for each training document based on the true labels using the label cluster assignment matrix (bottom).

Pre-computed clustering. We compute label clusters using K-Means algorithm, as depicted in Figure 4. We first obtain label features using average embedding of positive documents in a train set, following Chang et al. (2020), and compute the label-cluster assignment matrix. Label clusters are assigned to each training document using this matrix and ground-truth labels, and used as a multi-cluster prediction layer for XLGen-BCL training or sequence of cluster IDs for XLGen-MCG training.

3.1 Baseline Fine-Tuning

Figure 3a shows our baseline XLGen which simply fine-tunes text-to-text Transformers, e.g., T5 (Raf-

fel et al., 2020) or BART (Lewis et al., 2020), as our encoder-decoder framework on XMC dataset.

- *Input*: **task prefix**: input text
- *Output*: A sequence of label texts

For encoding, we add a prefix token 'MultiLabel' to the input text to inform the task type. Then, the output labels are generated as a sequence of labels in decoding. The model is fine-tuned with cross-entropy loss ($\mathcal{L}_{xent}$) given the sequence of label texts. In practice, the order of labels in decoding significantly influences the model performance. Following Yang et al. (2018), we sort the target labels in decreasing order of the frequencies. We also investigate various ordering effects and their impact on performance in §5.1.

3.2 Fine-Tuning with Cluster Prediction

Figure 3b shows the fine-tuning of text-to-text with an additional multi-cluster prediction layer (`XLGen-BCL`). By doing so, we expect the model learns label similarities and hence biases itself to generate labels relevant to the given cluster.

- *Input*: **task prefix**: input text
- *Multi-Cluster Layer*: a vector of $v_1, ...v_k$ where $v_i = 1$ if i^{th} cluster c_i is a positive cluster; otherwise $v_i = 0$ (1 1 0 ... 1...)
- *Output*: A sequence of label texts

The multi-cluster prediction layer is a vector of 0 or 1 that corresponds to the assigned clusters of instance, and is trained using the sequence of the last layer's hidden states of the encoder with a binary cross-entropy loss, $\mathcal{L}_{bce}$. The final objective is as follows:

$$\mathcal{L}_{xmc-bcl} = \mathcal{L}_{xent} + \lambda \mathcal{L}_{bce} \quad (1)$$

where $\mathcal{L}_{xent}$ is a cross-entropy loss term for the original text-to-text framework and $\mathcal{L}_{bce}$ is a binary cross entropy loss term for the cluster layer. λ is a weighting parameter for controlling $\mathcal{L}_{bce}$, to be chosen by dev-set performance.

3.3 Fine-Tuning with Cluster Decoding

`XLGen-BCL` utilizes a cluster prediction only as an auxiliary task to improve representations for a label prediction, thus, predicted clusters are not used in inference. Figure 3c shows the third variant, `XLGen` with a multi-cluster generation (`MCG`), which leverages predicted clusters as additional input tokens so that the cluster information can be used in inference.

- *Input1*: **task prefix**: input text ; a sequence of positive cluster IDs (c_1 c_2 c_{11}..)

	$\lvert D_{trn}\rvert$	$\lvert D_{tst}\rvert$	$\lvert L_{seen}\rvert$	$\lvert L_{unseen}\rvert$
EURLEX-4K	15,449	3,865	2,473	155
AMZNCAT-13K	1,186,239	306,782	13,275	0
WIKI10-31K	14,146	6,616	21,060	991
WIKI-500K	1,779,881	769,421	498,152	917

Table 1: Data statistics of the benchmark datasets; the number of train examples ($|D_{trn}|$), number of test examples ($|D_{tst}|$), number of labels in both train and test set ($|L_{seen}|$), and number of labels only in test set ($|L_{unseen}|$), which is zero-occurred labels in Table 3.

- *Output1*: A sequence of label texts
- *Input2*: **cluster prediction prefix**: input text
- *Output2*: A sequence of positive cluster IDs (c_1 c_2 c_{11}..)

We add a sequence of cluster IDs to the input text so that cluster information can be used while training (Input1-Output1). On the other hand, we have a new task with a clustering prefix 'MultiCluster' appended to the input text and predicts the sequence of labels (Input2-Output2). In training, these two tasks are trained simultaneously. Note that in inference, the predicted cluster IDs are appended to Input1 for the final label generation.

4 Experiments

4.1 Experimental Setups

Datasets. We use four widely used XMC benchmark datasets; three large-scale datasets with 4K∼30K labels (EURLEX-4K,AMZNCAT-13K, and WIKI10-31K) and one very-large-scale datasets with 500K labels (WIKI-500K). See Table 1 for the detailed data statistics.

Baseline and `XLGen` Training. We compare `XLGen` with three state-of-the-art baselines in XMC tasks; AttentionXML (You et al., 2019), X-Transformer (Chang et al., 2020), and XR-Linear (Yu et al., 2022). Note that all baseline models partition labels using hierarchical clustering. See A.1 for the detailed setups of baselines. Note it is common to upscale scores by ensemble learning for XMC baselines. However, for a fair comparison, we do not use any ensemble models for `XLGen` and baselines.

We train `XLGen` with T5 because BART (Lewis et al., 2020) performs worse for our task as shown in Table 13. By default, we sort labels in the decreasing frequency order to provide the training target sequence and infer the labels by beam search with size 5. For `XLGen-BCL` and `XLGen-MCG`, cluster sizes are optimized by dev set performance.

	EURLEX-4K		AMZNCAT-13K		WIKI10-31K		WIKI-500K	
	Mic.	*Mac.*	*Mic.*	*Mac.*	*Mic.*	*Mac.*	*Mic.*	*Mac.*
XR-Transformer	39.1	12.3	64.0	17.0	21.4	2.8	30.5	7.8
XR-Linear	44.6	15.1	53.2	18.6	19.2	3.6	17.2	3.3
AttentionXML	59.9	24.9	70.1	30.0	37.3	4.6	53.6	21.0
XLGen-base	59.8	27.5	69.8	38.8	**37.6**	**9.9**	55.1	**35.0**
XLGen-BCL	**60.7**	**28.4**	70.0	37.7	**37.6**	9.8	**55.4**	33.5
XLGen-MCG	60.2	28.2	**71.8**	**46.4**	37.4	9.6	**55.4**	33.6

Table 2: Full label performance. We report micro-averaged (*Mic.*) and macro-averaged (*Mac.*) F1 scores.

	EURLEX-4K		WIKI10-31K		WIKI-500K	
	0-st	1-st	0-st	1-st	0-st	1-st
XR-Transformer	0.0	0.5	0.0	1.7	0.0	0.0
XR-Linear	0.0	1.1	0.0	2.3	0.0	0.1
AttentionXML	0.0	2.4	0.0	0.2	0.0	1.3
XLGen-base	3.2	3.5	2.9	**8.4**	22.5	24.1
XLGen-BCL	4.3	**4.1**	3.3	**8.4**	23.2	24.8
XLGen-MCG	**4.5**	2.7	**11.1**	8.1	**23.7**	**25.5**

Table 3: Macro-averaged F1 scores in tail labels, which never occurred (0-st) or occurred once (1-st) in train set.

	EURLEX-4K	WIKI10-31K	WIKI-500K
XR-Transformer	8.6	3.3	7.0
XR-Linear	8.8	2.7	2.8
AttentionXML	18.7	1.3	11.3
XLGen-base	18.8	7.7	31.6
XLGen-BCL	19.3	8.0	31.4
XLGen-MCG	**21.2**	**10.1**	**32.7**

Table 4: Macro-averaged F1 scores in PU setting (50% of label deficit ratio).

We get the input text embedding by averaging the last hidden states from the pre-trained T5 encoder since T5 model does not have a CLS token. See A.2 to check more details.

4.2 Evaluation Metrics

Following the prior work in XMC, we report F1 score (F@k) of top-k label probabilities as a supplementary metric in A.4. However, such ranking metrics are not applicable to label generation tasks since the generative model only output positive label texts *sequentially* and the order of generated labels does not align with the confidence of the label; in other words, the formerly generated labels do not need to be more confident than the latter ones. Thus, we use conventional multi-label classification metrics, like micro-averaged F1 score (*Mic.*) and macro-averaged F1 score (*Mac.*), as main evaluation metrics.

In principle, evaluating XMC task with the ranking format is not appropriate for most cases as it requires predicting the number of correct labels as well (Amigo and Delgado, 2022). We therefore select predicted labels only when the predicted score is greater than the threshold optimized from the validation set as in You et al. (2019).

4.3 Results

We compare performances of XLGen and baselines in full labels (Table 2), tail labels (Table 3), and PU data setting (Table 4). For tail label and PU setting, we do not include AMAZNCAT-13K as it does not have zero-occurred labels. The best scores are **bold** and the second best scores are underlined. See A.4 for the full scores on tail labels and PU setting.

Full label performance. In the evaluation with full benchmark sets, all the XLGen models show outperforming or competitive performance compared to the classification-based baselines. For macro F1 scores, XLGen models hugely outperform the baselines, which empirically represents that our approach is strong at predicting infrequent but correct labels. In other words, XLGen models are less biased to predicting frequent labels. Compared to XLGen-base, both XLGen-BCL and XLGen-MCG generally show better performance, which demonstrates the effectiveness of the cluster prediction as an auxiliary loss.

Tail label performance. We measure macro F1 scores only for tail labels which never or one-time occur in the train set. We find every baseline extremely suffers from the tail labels, while XLGen shows significant improvements, demonstrating the power of generative models for long-tail labels. Surprisingly, XLGen even predicts never-seen, zero-occurred labels, only inferred from the semantic meaning of the input text. Similarly to full label performance, XLGen-BCL and XLGen-MCG perform better than XLGen-base, indicating

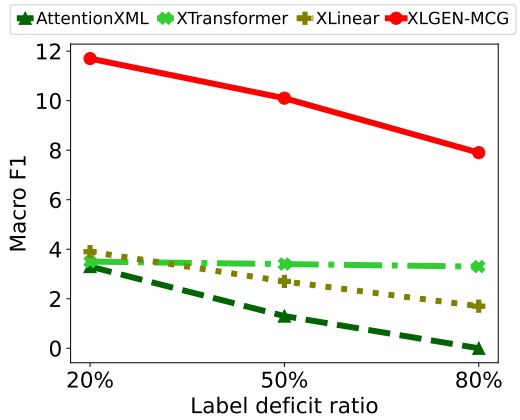


Figure 5: Macro-averaged F1 scores in PU setting on WIKI10-31K.

that guidance of label cluster improves tail label performance as well.

PU setting. In XMC, it is infeasible to annotate all relevant labels for an input text by checking every millions of labels. Therefore, many XMC datasets are indeed in PU setting. To evaluate the robustness against the positive and unlabeled properties, following Hu et al. (2021), we make PU data setting by randomly eliminating positive labels for each instance with 50% of deficit rate.

As XLGen is trained with fewer positive labels in PU settings, the generated output labels tend to be fewer as well, causing lower recall than expected. To increase the recall, we generate diverse label sequences using various sampling schemes in inference, which we call *ensemble generation*. We combine generated results from three decoding strategies; beam search with size 5, Top $P + K$ sampling, and sampling with 0.8 temperature.

We find XLGen models outperform the baselines. Specifically, XLGen-MCG shows significantly strong scores, which indicates having predicted clusters as an additional input helps predict infrequent but correct labels. In Figure 5, we additionally visualize macro F1 scores of PU settings on WIKI10-31K with various deficit rates. Although XLGen-MCG drops with an increasing deficit rate, it still shows significant gaps with baseline scores.

4.4 Feasibility of in-context learning in XMC

In-context learning (Brown et al., 2020) shows a potential of generating unseen but positive labels as depicted in Figure 1, such as "geyser" and "physical_reaction". In order to thoroughly validate the feasibility of in-context learning in XMC problems,

		EURLEX-4K		WIKI10-31K	
		Mic.	*Mac.*	*Mic.*	*Mac.*
XLGen-BCL		**60.7**	**52.4**	**37.7**	**20.0**
GPT-3	0-shot	9.2	6.3	7.6	4.5
	1-shot	17.2	14.7	20.3	13.4
	5-shot	15.7	10.4	23.5	16.6

Table 5: Label performance of XLGen-BCL and in-context learning settings on 100 randomly selected samples.

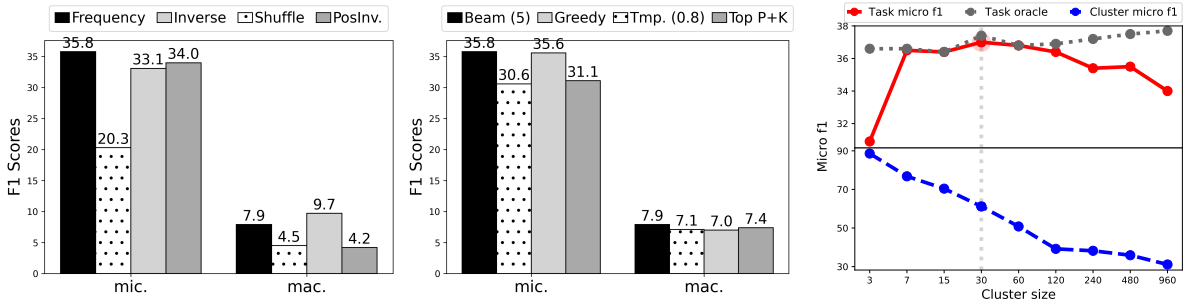
we select 100 samples randomly from EURLEX-4K and WIKI10-31K, and predict their labels using GPT-3 (Brown et al., 2020) in zero/one/five-shot setups. We explore a few variations of prompts by tweaking label order or selecting few-shot examples differently, and report the best scores in Table 5. The performance of in-context learning significantly improves when we use more examples in the prompt, but they are far from the performance of XLGen. Moreover, the performance gap between GPT-3 and XLGen is much larger in EURLEX-4K where labels are formally annotated than in WIKI10-31K where labels are annotated by random users without a solid guideline. Unlike other multi-label classification tasks, XMC treats an extremely large number of labels, making it difficult to predict most unseen labels based on a few examples in in-context learning. See A.3 for the details of the experimental setup for in-context learning and the performance comparison among prompt variations.

5 Ablation Study

We explore various factors that impact the performance of XLGen on WIKI10-31K, such as label orders (§5.1) and sampling strategies (§5.2) In order to reduce training costs, we mainly train XLGen-base on the base size model with epoch 5 for ablation tests. We then investigate the model performance by clustering sizes and algorithms (§5.3).

5.1 Label Orders

Label orders in decoder are important as XLGen sequentially generates labels. We compare three different label orders; label frequencies from high to low (Frequency), inverse label frequencies from low to high (Inverse), and shuffling where labels are randomly ordered per training epoch. Inspired by Lee et al. (2019), we also consider ignoring label orders by resetting positional embeddings of



(a) Label Order (b) Decoding Strategy (c) Cluster Size

Figure 6: Ablation study results on WIKI10-31K. (a) Performances of XLGen-base trained with various label orders. (b) Performances of XLGen-base trained with various decoding strategies. (c) Cluster sizes vs task and clustering performance. We also report oracle scores by using ground-truth cluster information in inference time.

each label as initial values in decoder [1] which we call label positional invariant setting (PosInv.).

Figure 6a shows task performance across different label orders. We find trade-offs between macro and micro F1 scores by the label frequency order (Frequency and Inverse) because inversely frequent label orders make the model generate long-tail labels earlier with certainty, thus, the scores of long-tail labels could improve. On the other hand, shuffling (Simig et al., 2022) crucially downgrades the performance since with randomly shuffled labels, XLGen tends to ignore co-occurrence patterns among labels in training time. Also, we conjecture that positional invariant setting does not work well as it tweaks the original positional embeddings of pre-trained T5 model.

5.2 Decoding Strategy

We now explore task performances with various sampling strategies in label generation. We compare greedy search, beam search, sampling with restrictions such as Top-K (Fan et al., 2018) and Top-P (Holtzman et al., 2020), and sharpening vocabulary distributions with a temperature parameter. In Figure 6b, we find that beam search with size 5 achieves the best scores. Interestingly, most sampling methods heavily degrade performances since our label spaces are not entirely open-ended. We also explore ensemble methods to combine label outputs from different sampling strategies. Unlike the PU setting, however, they are not helpful in the full data setup since a sufficient number of labels are already generated by a single best generation strategy. Find the Appendix A.6 for details.

[1] But we keep the position embeddings for token sequences in one label to learn token positions.

5.3 Cluster Strategy

We show the effect of clustering algorithms and their parameters. We train XLGen-MCG fine-tuning T5-base with epoch 5. We compare two clustering methods; K-means and Agglomerative clustering, and two text representations; TF-IDF and the recent T5 encoder. We find K-means and pre-trained T5 encoder shows the best performance over other combinations, as described in Appendix A.7.

Cluster size is another important factor for model performance. For example, a larger cluster size helps find label groups at a higher granularity, while it is much harder to be accurately predicted in inference time. Here, we choose cluster size to be a power of two on average (e.g., around 30 containing 1024 labels for WIKI10-31K on average). Figure 6c shows micro F1 scores of XLGen-MCG across cluster sizes in WIKI10-31K. Here we also report the upper bound of task performance (oracle) by using ground-truth cluster information. As we expect, clustering performance decreases as the cluster size increases since it is much harder to predict clusters in a larger cluster space. In terms of label prediction, we find that the model with smaller cluster sizes (e.g., ≤ 30) outperforms the larger ones, where the peak is around 30. Although a larger cluster size helps elaborately specify labels in the same category, lower cluster prediction performance harms label performances as well and leads to a bigger performance gap compared with oracle task scores.

6 Qualitative Analysis

Lastly, we evaluate the quality of generated labels via human evaluation (§6.1) and visualization of the semantic relations among labels (§6.2).

		AttentionXML		XLGen-BCL	
		#	%	#	%
Existing labels	*Correct*	674	39.2%	596	39.2%
	Wrong	776	45.1%	457	30.0%
	PU	270	15.7%	393	25.8%
New labels	*Correct*	0	0.0%	45	3.0%
	Wrong	0	0.0%	30	2.0%
Total		1,720	100%	1,521	100%

Table 6: Human evaluation on WIKI10-31K having 1,720 true labels. The predicted labels are annotated and categorized to *correct*, *wrong*, and *PU* labels, with their precision scores. Note that *correct* labels in the newly generated labels means they are possibly correct, according to the human annotators' decision.

6.1 Human Evaluation

The benchmark datasets have different label qualities. For example, the labels from EURLEX-4K, annotated by the Publication Office of EU, are refined and structured while Wiki datasets are collaboratively labeled by general users, so the quality of labels is relatively lower than the other benchmarks. Hence, we conduct human evaluations in both quantitative and qualitative ways to accurately measure the potential existence of PU labels and newly-generated labels. In particular, we randomly select 100 instances from the test set and extract incorrectly predicted labels and/or newly predicted labels by XLGen and baseline models. We then ask three human annotators to annotate and decide on possibly positive labels via majority voting.

Table 6 shows human evaluation results on the annotated WIKI10-31K. Note that the number of predicted labels by XLGen-BCL is less than true labels because XLGen does not generate label with low confidence. In AttentionXML, on the other hand, we choose top-K labels as many as the number of true labels for each instance, so it has the same total labels as true labels. Compared to the best baseline, AttentionXML, XLGen-BCL could generate more PU labels and reduce the number of wrong labels. Also, our method generates 75 (=45+30) newly generated labels out of 1,521 where 60% (=45/75) of them are correct, showing a relatively good generation quality of new labels. Of course, we can control our model to only count the candidate labels and not any of these new labels for more accurate predictions, as measured in Table 2.

Lastly, we provide annotation examples in Table 7. As we sort the label sequence by frequency in training for XLGen, frequently generated labels such as "wikipedia" or "wiki" are predicted first, followed by long-tail labels specified in the input text. For AttentionXML, on the other hand, top predicted labels seem more aligned with the input context, although frequently generated labels still come in front. Interestingly, new labels generated by XLGen come not only from the input context, but also previously generated labels. For instance, on the Wikipedia page of diet coke and mentos eruption, a new label "soda" is generated because input text contains "carbonated beverages" which is synonym of "soda". On the Wiki page of Vimeo, on the other hand, after XLGen generates the PU label "socialnetworking", followed by its synonyms such as "social_network" and "social_networking".

6.2 Label Semantics in XLGen

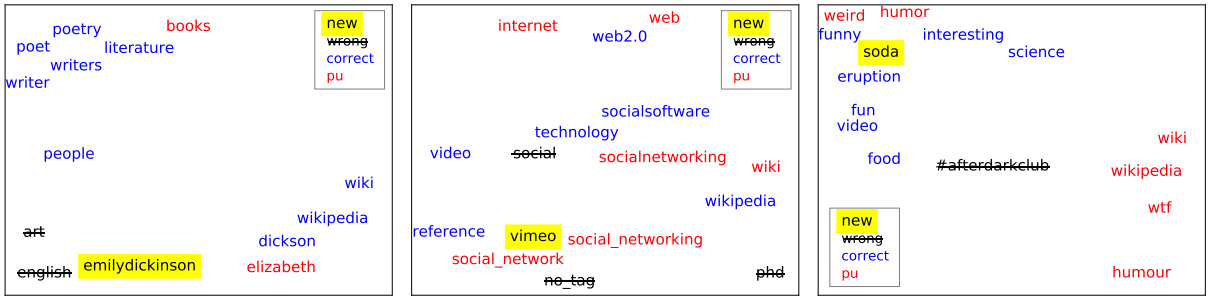
To better understand the semantics behind labels generated by XLGen, we visualize an annotated labels of three examples from Table 7 in Figure 7. We get label embeddings from the last hidden state of the fine-tuned XLGen-BCL decoder and project them into two-dimensional T-SNE (van der Maaten and Hinton, 2008). If a single label is split by multiple tokens, we average the last hidden layers of all tokens. We observe that frequently co-occurred labels (e.g., "wiki"-"wikipedia" or "weird"-"funny") have similar label embeddings. Also, the newly generated labels become close to the co-occurred labels (e.g., "soda" - "funny" or "eruption" in diet coke and mentos eruptions) via XLGen optimization.

7 Conclusion

We apply text-to-text Transformers to extreme multi-label classification, by tweaking the classification problem as generation of label texts. As we do not control the vocabulary space of generated labels, XLGen can create completely unseen but still relevant labels, inferred from the input context and semantic relationship from the previously generated labels. Our experiments show that XLGen outperforms the classification baselines in general, and significantly improves the long-tail performance and PU setting. Also, we observe utilizing label cluster information helps improve the performance in various settings. XLGen is expected to more benefit from pre-trained models as they become larger and powerful (Kaplan et al., 2020).

Input Document	Models	Labels
Emily Elizabeth Dickinson (December 10, 1830– May 15, 1886) was an American poet. Born in Amherst, Massachusetts to a successful family with strong community ties, she lived a mostly introverted and reclusive life. After she studied at the Amherst Academy for seven years in her youth, she spent a short time at ...	True	authors biography dickinson emily journal library literature openaccess people poem poet poetry reference research to-read wiki wikipedia writers
	AttentionXML	wiki poet writers wikipedia literature authors books ~~writing~~ ~~history~~ poets writer people poetry biography ~~inspiration~~ american poems ~~luule~~
	XLGen-BCL	wikipedia wiki people ~~art~~ books literature ~~english~~ poetry writers writer poet elizabeth dickinson emilydickinson
Screenshot of vimeo.com home page Vimeo is a video-centric social network site (owned by IAC/InterActiveCorp) which launched in November 2004. The site supports embedding, sharing, video storage, and allows user-commenting on each video page...	True	articles computer reference socialnetworks technology tools video web2.0 wikipedia
	AttentionXML	video web2.0 wikipedia wiki ~~media~~ ~~youtube~~ videos ~~videoblogging~~ ~~streaming~~
	XLGen-BCL	wikipedia wiki reference technology web internet ~~social~~ video web2.0 ~~no_tag~~ socialnetworking ~~socialsoftware~~ ~~phd~~ social_networking social_network vimeo
Diet Coke and Mentos Eruption is a reaction of Diet Coke and mint Mentos candies, a bottle of Diet Coke (other carbonated beverages may be used instead) and dropping some Mentos. This causes the Coke to foam at a rapid rate and spew into the air...	True	beverage candy chemistry coca-cola coke dietcoke drink eruption experiment experiments video explosion food fun funny interesting mint prank science
	AttentionXML	wikipedia fun science ~~diet~~ wiki funny coke ~~tv~~ video ~~health~~ interesting humor food
	XLGen-BCL	wikipedia wiki science interesting fun video funny food humor weird humour wtf ~~#afterdarkclub~~ soda eruption

Table 7: Ground-truth and predicted labels from XLGen-BCL and AttentionXML on input documents in WIKI10-31K. We ask human annotators to annotate labels to be correct (blue), wrong (~~strikethrough~~), and PU (red). For XLGen, we additionally mark potentially correct labels from the newly generated labels and their relevant contexts in input text with yellow box. (e.g., a possibly correct label soda is newly generated based on the fact that diet coke can be replaced with other carbonated beverage.)



(a) Emily Dickinson (b) Vimeo.com (c) Diet Coke and Mentos Eruptions

Figure 7: Visualization of generated labels by XLGen-BCL for the Wikipedia page examples in Table 7.

Limitations and Future Directions

First, we conduct our main experiments and additional analyses on certain languages such as English that has tremendous text corpora. Extension to the low-resource languages might be challenging since this work requires text2text pre-trained models where those languages are applicable (e.g., multilingual T5 model), as well as the corresponding XMC datasets.

Also, compared to the efficient classification baselines, generative models are relatively expensive in terms of memory and time. For example, our experiment requires a lot of training resource as pre-trained models have >200 millions parameters to be tuned. Thus, we use three p3.16xlarge AWS instances with 8 Nvidia V100 GPUs for training. Using more efficient version of Transformers (Tay et al., 2022) or applying distributed training should be considered for a resource reduction.

While in-context learning does not show comparable performance in XMC, we do observe that as the number of examples increases from zero to one to five, in-context learning can generate reasonable unseen but positive labels. It would be interesting to explore the potential of in-context learning in XMC with more advanced prompting and example sampling in the future.

Lastly, the XMC task has a risk of being biased or overfit to small training datasets (e.g., EURLEX-4K and WIKI10-31K contain only about 15,000

training examples). As with other commonly used NLP benchmarks, there is a potential risk that our proposed method may not work properly in the new test/train sets, though we anticipate that such a risk will be quite small.

Ethics Statement

We use the four XMC benchmark datasets which are publicly available and widely used in research [2]. The datasets with social tags (e.g., WIKI10-31K and WIKI-500K) may contain inappropriate vulgarisms if they are not filtered out from the original data processing.

References

Enrique Amigo and Agustín Delgado. 2022. Evaluating extreme hierarchical multi-label classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5809–5819, Dublin, Ireland. Association for Computational Linguistics.

Rohit Babbar and Bernhard Schölkopf. 2017. Dismec: Distributed sparse machines for extreme multi-label classification. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 721–729.

Peter F Brown, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–480.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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*, pages 3163–3171.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *ACL*, pages 889–898.

Eva L. Gibaja. 2015. A tutorial on multi-label learning. *ACM Computing Surveys*, 47:1–38.

N. Gupta, S. Bohra, Y. Prabhu, S. Purohit, and M. Varma. 2021. Generalized zero-shot extreme multi-label learning. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

[2] https://github.com/yourh/AttentionXML

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR*.

Wenpeng Hu, Ran Le, Bing Liu, Feng Ji, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2021. Predictive adversarial learning from positive and unlabeled data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7806–7814.

Himanshu Jain, Venkatesh Balasubramanian, Bhanu Chunduri, and Manik Varma. 2019. Slice: Scalable linear extreme classifiers trained on 100 million labels for related searches. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 528–536.

Atsushi Kanehira and Tatsuya Harada. 2016. Multi-label ranking from positive and unlabeled data. In *CVPR*, pages 5138–5146.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv:2001.08361*.

Joo-Kyung Kim and Young-Bum Kim. 2020. Pseudo labeling and negative feedback learning for large-scale multi-label domain classification. In *ICASSP*, pages 7964–7968.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, pages 3744–3753.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. 2017. Maximizing subset accuracy with recurrent neural networks in multi-label classification. *Advances in neural information processing systems*, 30:5413–5423.

Yashoteja Prabhu, Anil Kag, Shrutendra Harsola, Rahul Agrawal, and Manik Varma. 2018. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In *Proceedings of the 2018 World Wide Web Conference*, pages 993–1002.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits

of transfer learning with a unified text-to-text transformer. *JMLR*, 21:1–67.

Wissam Siblini, Pascale Kuntz, and Frank Meyer. 2018. CRAFTML, an efficient clustering-based random forest for extreme multi-label learning. In *International Conference on Machine Learning*, pages 4664–4673.

Daniel Simig, Fabio Petroni, Pouya Yanki, Kashyap Popat, Christina Du, Sebastian Riedel, and Majid Yazdani. 2022. Open vocabulary extreme classification using generative models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1561–1583, Dublin, Ireland. Association for Computational Linguistics.

Yukihiro Tagami. 2017. AnnexML: Approximate nearest neighbor search for extreme multi-label classification. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 455–464.

Yi Tay, Mostafa Dehghani, Jinfeng Rao, William Fedus, Samira Abnar, Hyung Won Chung, Sharan Narang, Dani Yogatama, Ashish Vaswani, and Donald Metzler. 2022. Scale efficiently: Insights from pre-training and fine-tuning transformers. In *ICLR*.

Che-Ping Tsai and Hung-Yi Lee. 2020. Order-free learning alleviating exposure bias in multi-label classification. In *AAAI*, pages 6038–6045.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605.

Marek Wydmuch, Kalina Jasinska, Mikhail Kuznetsov, Róbert Busa-Fekete, and Krzysztof Dembczynski. 2018. A no-regret generalization of hierarchical softmax to extreme multi-label classification. *Advances in neural information processing systems*, 31:6358–6368.

Yuanhao Xiong, Wei-Cheng Chang, Cho-Jui Hsieh, Hsiang-Fu Yu, and Inderjit Dhillon. 2022. Extreme Zero-Shot learning for extreme text classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5455–5468, Seattle, United States. Association for Computational Linguistics.

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.

Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. 2018. SGM: sequence generation model for multi-label classification. In *COLING*, pages 3915–3926.

Hui Ye, Zhiyu Chen, Da-Han Wang, and Brian Davison. 2020. Pretrained generalized autoregressive model with adaptive probabilistic label clusters for extreme multi-label text classification. In *International Conference on Machine Learning*, pages 10809–10819. PMLR.

Ian EH Yen, Xiangru Huang, Wei Dai, Pradeep Ravikumar, Inderjit Dhillon, and Eric Xing. 2017. Ppdsparse: A parallel primal-dual sparse method for extreme classification. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 545–553.

Ronghui You, Zihan Zhang, Ziye 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. *Advances in Neural Information Processing Systems*, 32:5820–5830.

Hsiang-Fu Yu, Prateek Jain, Purushottam Kar, and Inderjit S. Dhillon. 2014. Large-scale multi-label learning with missing labels. In *ICML*, pages 593–601.

Hsiang-Fu Yu, Kai Zhong, and Inderjit S Dhillon. 2022. Pecos: Prediction for enormous and correlated output spaces. *JMLR*, 23:1–32.

Jiong Zhang, Wei-cheng Chang, Hsiang-fu Yu, and Inderjit Dhillon. 2021a. Fast multi-resolution transformer fine-tuning for extreme multi-label text classification. *Advances in Neural Information Processing Systems*, 34:7267–7280.

Ximing Zhang, Qian-Wen Zhang, Zhao Yan, Ruifang Liu, and Yunbo Cao. 2021b. Enhancing label correlation feedback in multi-label text classification via multi-task learning. In *ACL Findings*.

Daoming Zong and Shiliang Sun. 2022. BGNN-XML: Bilateral graph neural networks for extreme multi-label text classification. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–12.

A Appendix

A.1 Details on Baseline Models

AttentionXML (You et al., 2019) is a label tree-based deep learning model. It uses a shallow and wide probabilistic label tree which allows to handle millions of labels and a multi-label attention mechanism by using raw text as input to capture the most relevant part of text to each label.

X-Transformer (Chang et al., 2020) is the first scalable approach to apply deep transformer models in XMC task. In particular, it uses a pre-trained transformer encoder to assign labels to corresponding cluster. For each hierarchical cluster level, OVA classifiers are trained by only using sample instances under the same cluster, called teacher forcing negative (TFN) strategy. Unlike AttentionXML which only uses negative sampling, X-Transformer also uses the negative instances positively predicted by the classifier from the previous cluster level, called matcher-aware negatives (MAN). Recently, Zhang et al. (2021a) proposed XR-Transformer to speed up X-Transformer's training time in recursive manner. Thus, we use XR-Transformer instead of X-Transformer for the comparison.

XR-Linear (Yu et al., 2022) has a very similar architecture with XR-Transformer, except that it only uses simple tf-idf text features instead of transformer encoder outputs. For OVA classification, linear matchers recursively solve XMC sub-problem for each hierarchical cluster level.

In order to fit score outputs into [0,1], we apply sigmoid post processor implemented by the authors for XR-Transformer and XR-Linear.

A.2 Details on `XLGen` Training

	XLGen-BCL	XLGen-MCG
EURLEX-4K	80	20
AMZNCAT-13K	80	20
WIKI10-31K	60	20
WIKI-500K	80	20

Table 8: Optimal cluster sizes for the `XLGen` training.

We finetune the T5-large (EURLEX-4K, WIKI10-31K) or the T5-base (AMAZONCAT-13K, WIKI-500K), with epoch 10 (EURLEX-4K, AMAZONCAT-13K) or epoch 5 (WIKI10-31K, WIKI-500K) based on the data and/or label size.

We set up input length as 500 for all benchmark datasets and use different output length based on the label lengths in train set; 90 for EURLEX-4K and 165 for other three benchmarks. We optimize `XLGen` using AdamW (Loshchilov and Hutter, 2019) with learning rate 2e-4.

	Sample	Label	EURLEX-4K		WIKI10-31K	
			Mic.	*Mac.*	*Mic.*	*Mac.*
0-shot	Random	Random	5.3	3.8	7.1	3.8
	Random	Frequency	**9.2**	**6.3**	**7.6**	**4.5**
	Most Label	Random	6.0	4.3	7.2	4.4
	Most Label	Frequency	5.0	3.5	6.7	3.5
1-shot	Random	Random	14.8	10.7	13.6	11.8
	Random	Frequency	16.1	10.4	**20.3**	13.4
	Most Label	Random	**17.2**	**14.7**	17.9	**16.8**
	Most Label	Frequency	15.1	12.2	18.1	16.1
5-shot	Random	Random	**15.7**	10.4	17.9	14.2
	Random	Frequency	13.1	9.7	**23.5**	16.6
	Most Label	Random	11.0	9.6	19.8	**18.0**
	Most Label	Frequency	12.5	**11.3**	21.5	15.1

Table 9: Micro-averaged and macro-averaged F1 scores in-context learning settings on 100 randomly selected samples. We test two label ordering strategies, random and decreasing label frequency (frequency), as well as two sampling strategies, random and selecting examples with the most labels (most label). The highest scores are **bold**.

For `XLGen-BCL`, we set up an initial weight value λ as 1.0 and reduce it to $\frac{1}{k}$ for every epoch number k.

For cluster-based `XLGen` architectures, we train k-means clustering and optimize the cluster size via cross-validation from the range of {10,20,30,...,100}. In Table 8, we report optimal cluster sizes for `XLGen` training.

Note that each of T5 models have 220 million (T5-base) or 770 million (T5-large) parameters to be tuned. Also for training, we use a small batch size (1) since pre-trained T5 models are large to be fitted in a single GPU machine. Due to the model size, we use two GPU machines via model parallelism for T5-large and a single GPU machine for T5-base in training. Also, due to the training cost and time, we report the performance scores from the single running of training and inference. We basically modify the T5 code from huggingface library [3], and our code will be publicly available at `https://github.com/alexa/xlgen-eacl-2023`.

A.3 In-context learning in XMC

For in-context learning, we use OpenAI GPT-3 text-davinci-002 model with temperature 0.7 and max tokens 256. To find the optimal prompt, we use prompt variations with different label orders

[3] `https://huggingface.co/`

	EURLEX-4K	AMZNCAT-13K	WIKI10-31K	WIKI-500K
	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10	F@1/F@3/F@5/F@10
XR-Transformer	**27.4**/47.5/47.2/34.7	**31.6**/55.2/53.5/38.0	**8.8**/20.3/24.4/23.8	24.1/34.2/32.8/25.5
XR-Linear	26.1/47.8/51.1/41.2	30.5/55.8/58.6/45.7	8.5/19.2/24.9/29.3	22.8/32.5/31.3/24.4
AttentionXML	26.9/**51.9**/**58.8**/59.9	30.8/**59.2**/67.0/ 69.9	8.6/**21.0**/**28.1**/**35.6**	**24.4**/42.9/48.9/52.8
XLGen-base	21.3/46.9/57.8/59.8	30.8/57.4/65.3/69.0	8.1/15.5/21.7/30.1	23.7/**44.0**/50.4/54.6
XLGen-BCL	21.2/47.2/58.7/**60.7**	31.0/57.7/65.5/69.2	8.1/15.4/21.5/30.1	23.8/43.8/50.3/54.6
XLGen-MCG	20.9/47.0/58.1/60.2	31.2/58.8/**67.5**/**71.2**	8.1/15.6/22.2/31.2	23.8/**44.0**/**50.5**/**54.8**

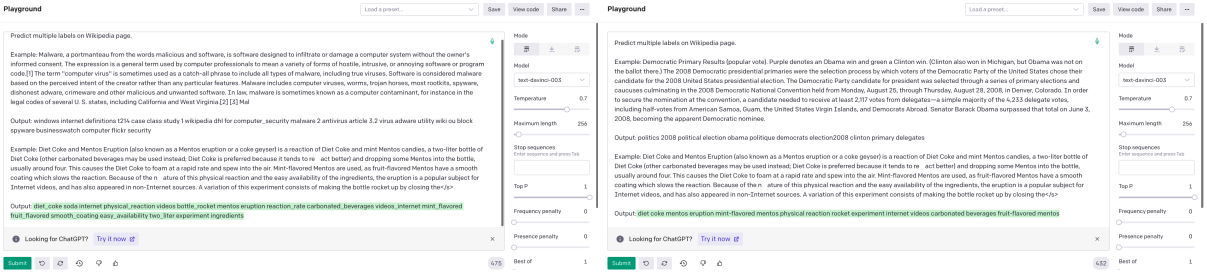
Table 10: Supplementary scores on benchmark datasets. We report ranking-based @k (k=1,3,5,10) F1 scores (F@k) as supplementary metrics. The highest scores are **bold**.

	EURLEX-4K			WIKI10-31K			WIKI-500K		
	0-shot	1-shot	5-shot	0-shot	1-shot	5-shot	0-shot	1-shot	5-shot
XR-Transformer	0.0/0.0	1.5/0.5	4.7/2.3	0.0/0.0	2.5/1.6	2.9/1.7	0.0/0.0	0.0/0.0	0.0/0.0
XR-Linear	0.0/0.0	5.9/1.1	8.6/2.7	0.0/0.0	2.8/2.3	2.9/2.4	0.0/0.0	0.2/0.1	1.4/0.9
AttentionXML	0.0/0.0	16.3/2.4	28.4/8.3	0.0/0.0	0.3/0.2	5.6/1.9	0.0/0.0	2.2/1.3	16.1/9.2
XLGen	5.3/3.2	21.4/3.5	34.9/10.8	4.5/2.9	14.4/**8.4**	**17.9**/**7.5**	**21.6**/22.5	35.6/24.1	39.6/28.7
XLGen-BCL	**7.3**/4.3	**25.0**/**4.1**	**36.1**/**11.4**	**5.0**/3.3	**14.5**/**8.4**	17.6/7.3	18.6/23.2	36.2/24.8	40.2/29.5
XLGen-MCG	**7.3**/**4.5**	16.7/2.7	35.9/11.1	**5.0**/**11.1**	13.9/8.1	17.3/7.2	20.7/**23.7**	**37.0**/**25.5**	**40.6**/**29.9**

Table 11: Task performances on benchmark datasets in full few-shot setup. We use conventional micro-averaged ($Mic.$) and macro-averaged ($Mac.$) F1 scores and mark $Mic./Mac.$ in the table. The highest scores are **bold**.

	EURLEX-4K			WIKI10-31K			WIKI-500K		
	Positive Unlabeled Deficit Ratio			Positive Unlabeled Deficit Ratio			Positive Unlabeled Deficit Ratio		
	20%	50%	80%	20%	50%	80%	20%	50%	80%
XR-Transformer	32.6/10.9	25.9/8.6	14.6/4.7	20.7/3.5	16.7/3.4	12.0/3.3	29.2/8.0	24.5/7.0	17.7/5.1
XR-Linear	38.6/12.1	27.8/8.8	12.5/3.7	16.1/3.9	11.5/2.7	5.1/1.7	14.4/3.7	9.9/2.8	5.9/2.0
AttentionXML	**57.6**/23.6	**52.3**/18.7	**40.0**/11.1	**34.7**/3.3	27.7/1.3	14.1/0.0	50.9/18.9	46.0/11.3	35.5/5.0
XLGen-base	55.7/24.4	47.5/18.8	31.3/9.8	33.6/9.1	32.3/7.7	22.7/2.8	**51.1**/**37.3**	**48.7**/31.6	**37.9**/**25.4**
XLGen-BCL	56.0/24.4	47.9/19.3	31.4/10.0	33.2/9.1	**33.0**/8.0	23.6/2.9	50.8/37.0	48.5/31.4	37.8/25.2
XLGen-MCG	55.5/**27.8**	48.2/**21.2**	32.6/**13.3**	32.4/**11.7**	**33.0**/ **10.1**	**24.0**/**7.9**	50.7/**37.3**	48.5/**32.7**	35.9/24.8

Table 12: Task performances on benchmark datasets in full PU setup. We use conventional micro-averaged ($Mic.$) and macro-averaged ($Mac.$) F1 scores and mark $Mic./Mac.$ in the table. The highest scores are **bold**.



(a) Label frequency order with random sample (b) Label random order with random sample

Figure 8: Input prompt and generated outputs (green-shaded) for Wikipedia page of Elizabeth Dickinson with 1-shot example.

and few-shot example sampling strategy. Furthermore, we test two label ordering strategies, random and decreasing label frequency, as well as two sampling strategies, random and selecting examples with the most labels. See Figure 8 for GPT-3 prompt input and generated output. Table 9 shows the in-context learning performances across different label ordering and sampling strategies. The best macro-averaged F1 scores for WIKI10-31K are achieved with label frequency ordering with random sampling; however, there is no consistently outperforming strategy for EURLEX-4K.

A.4 Additional Task Performances on Benchmark Datasets

For the full setup, we also report ranking based scores in Table 10. In general, for supplementary metrics (F@k) XLGen shows comparable results with baselines except F@1, and F@3 in EURLEX-4K and WIKI10-31K. Note that for XLGen, we just treat the order of generated labels as a rank, which might **not** be correct since such generated labels should have a equal priority in theory. For this reason, XLGen has lower F@k scores with smaller k. However, such score gaps between baselines and XLGen decrease as k increases, like EURLEX-4K with XLGen-BCL, or even XLGen achieves higher performances in the larger benchmarks (e.g., F@5 and F@10 in AMZNCAT-13K and WIKI-500K). Also, full micro/macro F1 scores for tail labels and PU settings are in Table 11 and Table 12, respectively.

A.5 Additional Analyses on Base Model Comparison

XLGen-base	EURLEX-4K		WIKI10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
T5-base	**58.0**	**23.8**	**35.8**	**7.9**
BART-base	55.6	23.4	35.0	7.7

Table 13: Task performances of XLGen-base trained with different pre-trained model architectures on EURLEX-4K and WIKI10-31K. The highest scores are **bold**.

For XLGen, we can use any pre-trained text-to-text models. We compare task performance of two popular text-to-text models in Table 13; T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) by finetuning XLGen-base. In general T5 model outperforms BART, therefore, we use pre-trained T5 architectures for our main experiments.

A.6 Additional Analyses on Decoding Strategy

Followed by Figure 6b, Table 14 shows a task performance across various decoding strategies, including different beam size for beam search and a single sampling restriction.

Additionally, instead of choosing single generation strategy, we can even consider to integrate generation outputs from different generation strategies. For ensemble generations, we choose three single generation strategies; beam search with size 5, Top $K + P$ sampling and sampling with temperature 0.8 to get diverse label sequences. We also consider two different types of joining method; inner join to union all labels and outer join to intersect labels from single generations.

	EURLEX-4K		WIKI10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Greedy	57.5	23.3	35.6	7.0
Beam (3)	**58.0**	23.7	35.7	7.8
Beam (5)	**58.0**	**23.8**	**35.8**	**7.9**
Beam (10)	57.9	**23.8**	35.4	7.8
Tmp. (0.8)	53.7	21.9	30.6	7.1
Top-K (50)	51.7	20.8	28.7	6.8
Top-P (0.9)	53.2	21.1	28.8	6.5
Top $P + K$	53.6	21.5	31.1	7.4

Table 14: Performances of XLGen-base trained with different decoding strategies.

XLGen-base	EURLEX-4K		WIKI10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Beam (5)	**58.0**	23.8	**35.8**	7.9
Ens. Outer	53.5	**24.5**	31.5	**10.0**
Ens. Inner	54.0	18.9	29.2	4.2

Table 15: Task performance of XLGen-base from the best single stratagy (beam search with size 5) and ensemble generations on EURLEX-4K and WIKI10-31K. The highest scores are **bold**.

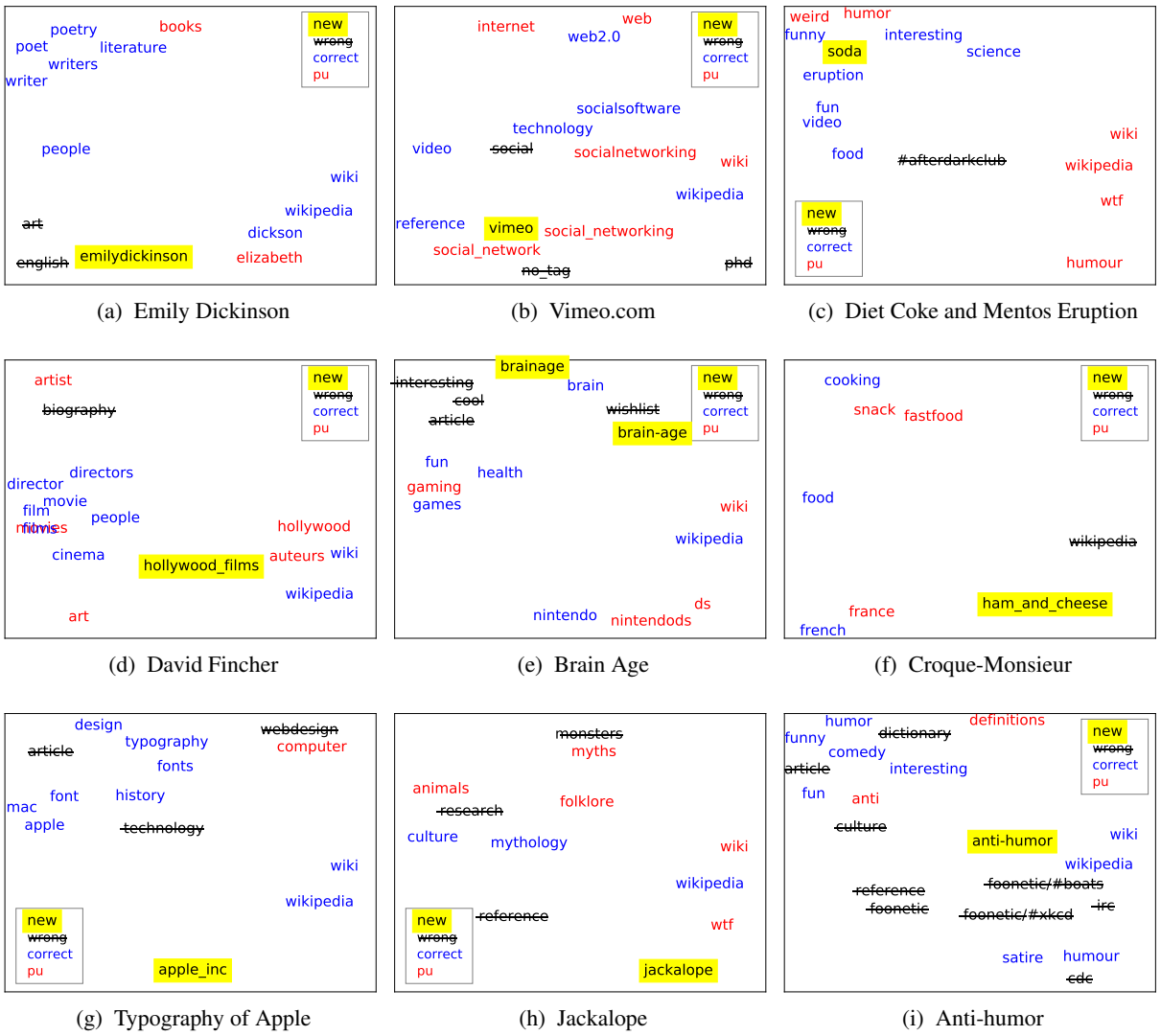
Table 15 shows task performance of ensemble generations. We find that outer joining ensemble generation could improve macro F1 scores as it includes more labels than single result. However, it simultaneously drops other micro F1 scores due to the high chance to contain wrongly predicted labels as well. On the other hand, inner joining ensemble generation in general harms the performance by restricting predicted labels occurring at any single generations, though this yields higher micro F1 scores than inner joining ensemble results.

A.7 Additional Analyses on Clustering and Representation

XLGen-base	EURLEX-4K		WIKI10-31K	
	Mic.	*Mac.*	*Mic.*	*Mac.*
Kmn. + tf-idf	58.4	24.1	36.6	**8.9**
Kmn. + t5-enc.	**58.5**	23.9	36.8	8.8
Ahcl. + tf-idf	58.4	**24.4**	36.8	8.8
Ahcl. + t5-enc	58.0	24.0	**37.0**	8.6

Table 16: Task performances trained with different cluster algorithm and input features on EURLEX-4K and WIKI10-31K. Here we fix cluster size as 30. The highest scores are **bold**.

We compare two clustering algorithms; K-means and Agglomerative hierarchical clustering, and two text representations for the label features; TF-IDF and the last hidden states of T5 encoder in Table 16. We find that both algorithms show comparable per-



(a) Emily Dickinson (b) Vimeo.com (c) Diet Coke and Mentos Eruption

(d) David Fincher (e) Brain Age (f) Croque-Monsieur

(g) Typography of Apple (h) Jackalope (i) Anti-humor

Figure 9: Visualization of generated labels by XLGen-BCL for Wikipedia pages of annotated examples in Table 17.

formances. As computing cost is more expensive in Agglomerative hierarchical clustering, we mainly use K-means in our experiments. For text representation, the pre-trained T5 encoder achieves similar or slightly better performance to TF-IDF vectors. Pre-trained T5 encoder is more efficient in training as it has much lower size of dimensionality (e.g., 768 in t5-base) than tfidf (e.g., >100,000 for both EUR-LEX and WIKI10-31K). Thus, for all experiments with clustering method, we use K-means with pre-trained T5 encoder text representation.

A.8 Additional Examples of Human Annotation

In Table 17, We provide more annotation examples from WIKI10-31K, following the Table 7 to show how XLGen generates labels. In Figure 9, we also provide visualizations of generated labels by XLGen for examples in Table 17.

Input Document	Models	Labels
Emily Elizabeth Dickinson (December 10, 1830– May 15, 1886) was an American poet. Born in Amherst, Massachusetts to a successful family with strong community ties, she lived a mostly introverted and reclusive life. After she studied at the Amherst Academy for seven years in her youth, she spent a short time at ...	True	authors biography dickinson emily journal library literature openaccess people poem poet poetry reference research to-read wiki wikipedia writers
	AttentionXML	wiki poet writers wikipedia literature authors books ~~writing~~ ~~history~~ poets writer people poetry biography ~~inspiration~~ american poems ~~luule~~
	XLGen-BCL	wikipedia wiki people ~~art~~ books literature ~~english~~ poetry writers writer poet elizabeth dickinson emilydickinson
Screenshot of vimeo.com home page Vimeo is a video-centric social network site (owned by IAC/InterActiveCorp) which launched in November 2004. The site supports embedding, sharing, video storage, and allows user-commenting on each video page...	True	articles computer reference socialnetworks technology tools video web2.0 wikipedia
	AttentionXML	video web2.0 wikipedia wiki ~~media~~ ~~youtube~~ videos ~~videoblogging~~ ~~streaming~~
	XLGen-BCL	wikipedia wiki reference technology web internet ~~social~~ video web2.0 ~~no_tag~~ socialnetworking ~~socialsoftware~~ ~~phd~~ social_networking social_network vimeo
Diet Coke and Mentos Eruption is a reaction of Diet Coke and mint Mentos candies, a bottle of Diet Coke (other carbonated beverages may be used instead) and dropping some Mentos. This causes the Coke to foam at a rapid rate and spew into the air...	True	beverage candy chemistry coca-cola coke dietcoke drink eruption experiment experiments video explosion food fun funny interesting mint prank science
	AttentionXML	wikipedia fun science ~~diet~~ wiki funny coke ~~tv~~ video ~~health~~ interesting humor food
	XLGen-BCL	wikipedia wiki science interesting fun video funny food humor weird humour wtf ~~#afterdarkclub~~ soda eruption
David Leo Fincher (born August 28, 1962) is an Academy Award-nominated American filmmaker and music video director known for his dark and stylish movies such as Seven, Fight Club, Zodiac and The Curious Case of Benjamin Button...	True	cinema david director directors figures film filmmaking films fincher inspiration movie people wiki wikipedia
	AttentionXML	wiki directors video wikipedia cinema ~~pitt~~ people director films movies movie film filmmaker ~~brad~~
	XLGen-BCL	wikipedia wiki people art film ~~biography~~ movies artist movie cinema films director directors auteurs hollywood hollywood_films
Brain Age: Train Your Brain in Minutes a Day!, also known as Dr. Kawashima's Brain Training: How Old Is Your Brain? in PAL regions, is an entertainment video game that employs puzzles. It was developed and published by the video gaming company Nintendo for the Nintendo DS handheld video game console...	True	@mentat biology brain braintraining computer exercise fitness fun game games health medical nintendo read science sudoku unit4 wikipedia
	AttentionXML	wikipedia game games fun science nintendo sudoku brain ~~mind~~ ds wiki ~~video~~ ~~memory~~ gaming puzzle puzzles videogames nds
	XLGen-BCL	games wikipedia fun health brain nintendo nintendods wiki gaming ~~wishlist~~ ~~article~~ ~~interesting~~ ~~cool~~ ds brain-age brainage
A croque-monsieur is a hot ham and cheese (typically emmental[citation needed] or gruyère) grilled sandwich. It originated in France as a fast-food snack served in cafés and bars ...	True	cooking food french recipe sandwich
	AttentionXML	food ~~wikipedia~~ cooking french ~~wiki~~
	XLGen-BCL	~~wikipedia~~ food france french cooking ham_and_cheese fastfood snack
Typography of Apple Inc. refers to Apple Inc.'s use of typefaces in marketing, operating systems, and industrial design. Apple has used three corporate fonts throughout its history: Motter Tektura, Apple Garamond and Adobe Myriad. For at least 18 years, Apple's corporate font was a custom variant of the ITC Garamond typeface, called Apple Garamond ...	True	adobe apple branding chronology computer computers design design.fonts fmp font fonts helpful history imac ipod list mac macintosh marketing myriad print pro product reference sda spunti storia typography wiki wikipedia
	AttentionXML	typography fonts apple font design wikipedia type typeface wiki tipografia ~~ttf~~ macintosh reference mac history graphics ~~logo~~ graphic ~~designers~~ ~~webdesign~~ ~~graphicdesign~~ ~~diseño~~ computer typographer ~~ipod~~ ~~brand~~ ~~article~~ ~~technology~~ ~~business~~ ~~advertising~~
	XLGen-BCL	wikipedia wiki history ~~article~~ design ~~technology~~ computer ~~webdesign~~ mac apple typography fonts font apple_inc
The jackalope — also called an antelabbit, aunt benny, Wyoming thistled hare or stagbunny — is an imaginary animal of folklore and a supposed cross between a jackrabbit and an antelope, goat, or deer, which is usually ...	True	american animal creatureproject cryptozoology culture fiction humor humour myth mythology storyideas wikipedia
	AttentionXML	folklore wikipedia animals mythology wiki cryptozoology culture ~~monsters~~ animal weird interesting myth
	XLGen-BCL	wikipedia wiki ~~reference~~ ~~research~~ culture mythology animals folklore wtf myths ~~monsters~~ jackalope
Anti-humor and anti-jokes[1] (also known as unjokes) are a kind of humor based on the surprise factor of absence of an expected joke or of a punch line in a narration which is set up as a joke. This kind of anticlimax is similar to that of the shaggy dog story.[2] In fact, John Henderson sees the "shaggy dog story" ...	True	comedy favourites fun funny humor humour information interesting jokes people postmodernism wiki wikipedia
	AttentionXML	humor wikipedia funny comedy humour fun satire wiki ~~dog~~ ~~animals~~ ~~standup~~ ~~parody~~ joke
	XLGen-BCL	wikipedia wiki ~~reference~~ interesting ~~article~~ ~~culture~~ fun funny humor ~~irc~~ ~~foonetic~~ ~~foonetic/#xked~~ definitions ~~dictionary~~ humour comedy satire anti ~~cdc~~ ~~foonetic/#boats~~ anti-humor

Table 17: Additional examples of ground-truth and predicted labels in WIKI10-31K, following Table 7.