A Model for Zero-shot Text Multi-labeling Using Semantics-based Labels

Dan Dickinson
American Family Insurance
Madison, USA
ddickins@amfam.com

Ananth Raj GV
American Family Insurance
Madison, USA
agauribi@amfam.com

Glenn Fung
American Family Insurance
Madison, USA
gfung@amfam.com

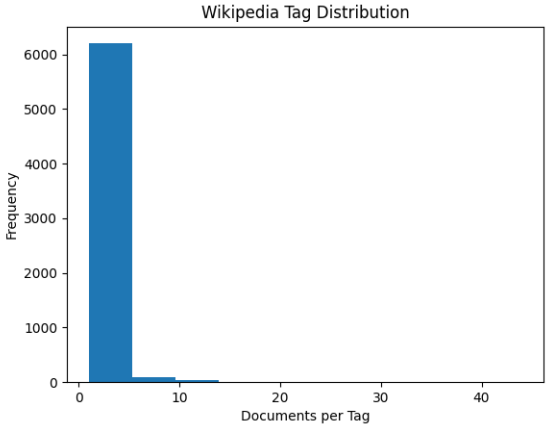
***Abstract*—We introduce a transformer-based method to associate relevant tags to text passages or blocks such as categories to pages of a website, marking sections in an article, or social postings subject tagging. In contrast with traditional multi-label formulations, the proposed approach uses semantic definitions of the tags available during training, and the model outputs a binary prediction of whether the described category applies to a document or not. The transformer-based model learns the semantics of the definition of a tag, and therefore works for tags not seen during training. Performance on domain-specific datasets can be further improved via transfer learning after fine-tuning with relatively little additional labeled data required.**

I. Introduction

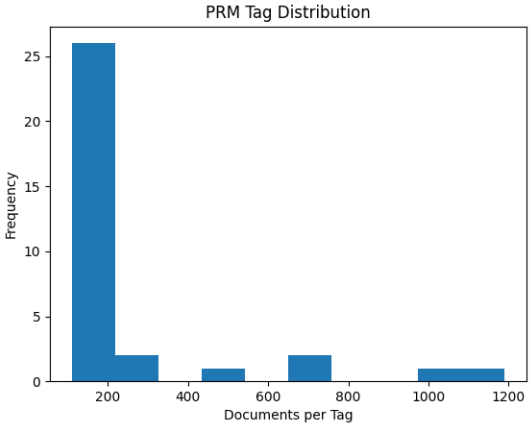
Most modern websites, content management systems, and other document storage systems allow blocks of text to be "tagged" with categories or concepts that apply to the content of the passage. This paper introduces a method that automatically determines which tags are relevant to which pages, and allows for adding additional tags without labeled examples. Using the language and notation of graph theory, this becomes an exercise in link prediction in bipartite graphs that allows for adding nodes in either or both subsets (text blocks and tags) [1]. We are building the model for deployment in our company's centralized knowledge graph, where tags play a crucial role in retrieving the most relevant information, and both new content and categories are added frequently as we ingest information from the variety of internal systems.

With sufficient domain-specific labelled examples and a fixed set of tags, it is a simple matter to build a multi-label classifier or multiple tag-specific classifiers to tag any existing or new pages. In real life one of the challenges with this standard approach is that it is common for the distribution of the tags available for training to be heavily right-skewed making it hard to learn how to correctly assign tags for which there are few examples [2], [3]. As an example, figure 1 shows the distributions of tags in the training data for the two datasets we used for experiments in this paper.

Another challenge with the standard multi-label approach occurs when a new, previously unseen tag is considered. A multi-label model will need to be re-trained to accommodate the new class and therefore require additional labeled examples of pages, meaning the training data will grow and become more cumbersome as new tags become relevant. To alleviate the need for lots of domain-specific labeled data and to accommodate new tags without retraining, our method takes advantage of the power and flexibility of modern transformer-based models.



(a) Wikipedia Tag Distribution



(b) PRM Tag Distribution

Fig. 1: Tag Distributions. Most tags are associated with relatively few documents. For the Wikipedia dataset, the overwhelming majority of tags have fewer than 5 instances to train on, and in the PRM most tags have fewer than 200 training examples.

Our proposed tagging approach does not use the tags simply as labels but also considers the semantics of the corresponding definitions provided at training time. As far as we know this is the first time that the problem of document tagging as been approached this way using transformers. An approach has been recently proposed for tagging sentences using a similar architecture, however the work focuses on short passages or sentences and the labels are defined by one or two tokens only [4]. Our approach is similar to that of extractive question answering models, like those trained on the SQuAD dataset [5], where the labeled set contains question, document, answer-span triplets (q, d, a). Inputting the question and document into the model simultaneously allows tokens from each to attend to the other and learn the semantics of asking a question. This approach increases performance and allows for both unseen questions and unseen documents to be used. Our approach and goals are similar: we provide additional context to the model to improve learning efficiency, learn the semantics of tagging, and apply to unseen categories and passages. Further discussion of the influence and motivation provided by current QA and zero- and few-shot approaches is deferred to section V.

In summary, the main contributions of this work are:

1) We propose a text passage tagging framework that addresses the practical limitation of standard multi-label classification approaches in the presence of long-tailed label distributions i.e many classes with few examples.
2) Our approach can predict unseen tags (zero-shot learning) with high accuracy given that a sufficient semantic definition of the tags are provided.

The rest of our paper is structured as follows: section II lays out the notation and graph concepts used throughout the paper, section III details the model structures we use, experiments are covered in section IV, section V covers background and related work, and conclusions and ideas for future work are detailed in section VI.

II. Notation and Problem Setting

A. Notation

We use notation from graph theory throughout the remainder of the paper. Though we do not use any machinery from graph theory, we find the notation helpful for several reasons: it is expressive and succinct, it helps to emphasize our approach to the problem as binary classification rather than multi-label, and it highlights the easy integration and end application of our proposed models into our corporate knowledge graph.

Let G be a bipartite directed graph with nodes N and edges E. For $e \in E$, let e_h and e_t denote the head and tail of the edge respectively. Let $C, D \subset N$, $C \cap D = \emptyset$, $C \cup D = N$ its two components with $e_t \in C$ and $e_h \in D$ for all $e \in E$.

B. Tagging as Link Prediction

In our case, the nodes in C consist of tags or categories for content, and D consist of text content. For example `homeowners, condo, renters` are the categories or tags represented by nodes in C, and the text from web pages about auto insurance, life insurance, liability coverage for home policies are documents represented by nodes in D. We say a tag in C `APPLIES TO` a given piece of content from D. The edges in E represent the `APPLIES TO` relationship, and since no tags apply to other tags, or content to other content, we have a bipartite graph.

Using the graph setup specified above, we seek to construct a model to predict the links, or edges representing `APPLIES TO`, between nodes in C representing categories, and nodes in D representing pages or documents. Hence, instead of learning a multi-label classifier $f(d) \rightarrow [\delta_{c_1}, ..., \delta_{c_k}]$ where δ_c is 1 if c `APPLIES TO` d and 0 otherwise, that predicts links to a particular tag, we aim to learn $f(c, d) \rightarrow \{0, 1\}$ that can be used to determine `APPLIES TO` between any $c \in C$ and $d \in D$.

C. Deployment Considerations

One potential drawback to using a binary classifier rather than traditional multi-label approach is speed; rather than getting all predicted tag links for a given document in a single call, we must run $f(c_i, d) \rightarrow \{0, 1\}$ for all $c_i \in C$. In most applications, however, tags are not updated in real-time, so getting inferences from the model can be batched and run during off-hours. In our corporate knowledge graph application we can simply perform inference when content is ingested, and make updates - if necessary - overnight.

III. Model Structure

A. General Structure

We specify several models in III-B, III-C, and III-D, all of which differ only in the final few classification layers; they all share a common structure for input data and majority of their architecture. In this section we lay out the general structure of the models and input data and provide further details of each variant in the following subsections.

We use a transformer-based (specifically BERT [6]) model, with text from a tag node $c \in C$ as the first sequence, and text from a content node $d \in D$ as the second sequence. Using the special tokens as defined in [6], that is:

`[CLS]` <*c*> `[SEP]` <*d*> `[SEP]` (1)

where ¡c¿ are the tokens from the tag c and ¡d¿ are the tokens from the document d. The transformer model output is fed into further layers (as specified in the remainder of this section) and finally a classification head to predict whether there is an edge $e \in E$ with $e_t = c$ and $e_h = d$.

Modern BERT-like NLP models make extensive use of the context in which a word is used, so rather than just using the one or two-word tags from C as in [4], we use a brief semantic definition or description of the tag to provide additional context. For example, instead of using "renter's insurance", we might use "Insurance providing coverage for liability and property losses stemming from renting a residence". The input into the model is then:

`[CLS]` <$desc(c)$> `[SEP]` <*d*> `[SEP]` (2)

where $desc(c)$ denotes the tokens of the description of category $c \in C$.

While getting a description of each category $c \in C$ is additional data that must be gathered, it is only required of the categories. Categories are generally much smaller in number than documents, their descriptions can often be pulled from a corporate glossary, and the addition of a description can be enforced at the time a category is created in most popular content management systems, which make collecting and maintaining the additional data a low cost effort.

Since all inputs to the model are unstructured text, the architecture allows us to add new tags and their descriptions or content nodes to the graph and determine which `APPLIES TO` relationships should be added, without requiring additional labeled examples. In addition, by inspecting the attention weights in the model, we can see where the model "learns" any tag, even those not seen during training. While such findings do not serve as proof of the method, they are invaluable in a business context and serve as a great way to explain the predictions and increase belief in the utility of the model. Table I shows the top attention weights for the `[CLS]` token of the fourth layer in BERT where the model learned the important semantic parts of the tag. In both tables, neither the description nor document were seen during training. The following subsections detail the various final classification layers we tested to make relationship predictions.

B. Document-level Model Structure (DM)

Our baseline and simplest architecture we tested, shown in figure 2, was a standard binary classification head on top of the BERT-based architecture specified in section III-A. We used the short description of the category and truncated the document (if necessary) so that the total number of input tokens was 512 or fewer. The output from BERT corresponding to the `[CLS]` token was used as input to a single feed-forward layer with output dimension one and a sigmoid activation. Figure 2 shows a diagram of the document-level model structure.

Anecdotally, many documents start with an introduction or overview that tend to capture the categories relevant to the page well, and the results in IV-B show the strength of this combination of architecture and input. However, due to the truncation required, it does not capture information beyond the beginning of a document. In longer documents, which tend to have more relevant categories due to their length and therefore require more information, truncation can lead to ignoring a significant amount of context.

C. Paragraph-level Model Structure (PM)

To alleviate the issue of truncating documents and ignoring information, we broke each input document into paragraphs (p_i) and derived paragraph-level targets from the page using the assumption that if a tag applied to a page, it applied to all paragraphs within. We trained paragraph-level classifier using the same architecture as III-B, then during evaluation ran each paragraph in a document through the model resulting in a vector of predictions for each document. To make a

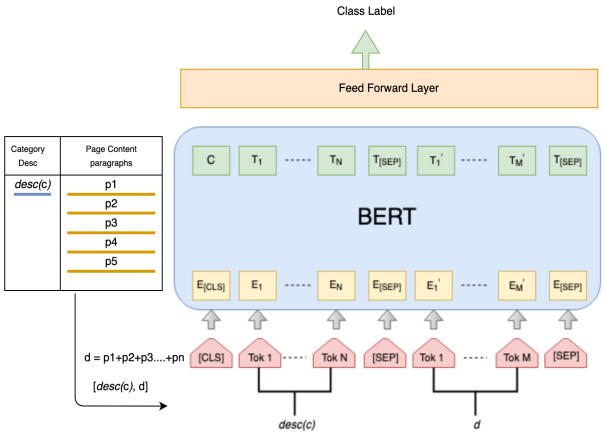


Fig. 2: Document Model (DM)

document-level prediction, we simply took the maximum of the paragraph predictions. By aggregating predictions to the document level only at evaluation time we only require one tag-paragraph pair at a time during training but can still take advantage of all tokens in the document during evaluation. Figure 3 shows a diagram of the paragraph model structure.

While this architecture does capture all information in the document, the derived labels for any given paragraph might not be accurate, and each section is treated independently and therefore can not attend to tokens in other sections.

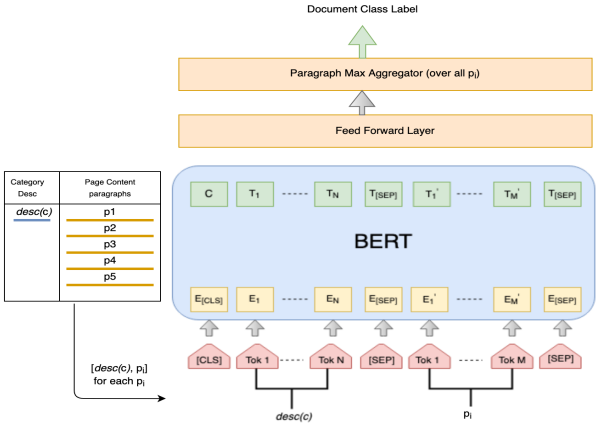


Fig. 3: Paragraph Model (PM)

D. MIL Model Structure (MIL)

A model that attends to tokens in other sections of the same document will have the benefit of using the entire context to make predictions and eliminate the need for derived paragraph labels. To do so, we used an architecture based on Multiple Instance Learning, where bags (documents) are comprised of multiple instances (paragraphs p_i), and labels are only known for a bag. For a given document, all tag-paragraph pairs are run through the same architecture as III-C. The output corresponding to the `[CLS]` token from each is stacked to form a matrix, which is run through MIL pooling as in [7].

Token	Index	Weight	Sequence
[CLS]	0	0.219	desc
[SEP]	109	0.085	desc
[SEP]	511	0.071	doc
sport	16	0.024	desc
hoop	7	0.015	desc
basketball	1	0.010	desc
rectangular	34	0.008	desc
defender	61	0.008	desc
team	15	0.007	desc
compete	37	0.006	desc
hoop	64	0.005	desc
teams	20	0.005	desc
mvp	404	0.004	doc
is	13	0.004	desc
the	60	0.004	desc
players	26	0.004	desc

(a) basketball

Token	Index	Weight	Sequence
[CLS]	0	0.198	desc
[SEP]	69	0.084	desc
[SEP]	504	0.067	doc
soccer	10	0.020	desc
association	1	0.015	desc
rectangular	21	0.012	desc
.	68	0.012	desc
football	2	0.011	desc
football	8	0.007	desc
the	70	0.006	doc
team	58	0.006	desc
game	16	0.006	desc
opposition	43	0.006	desc
pitch	25	0.005	desc
opposing	54	0.005	desc
goals	64	0.005	desc

(b) association football

TABLE I: Token-level attention weights for semantic definitions of basketball and association football paired with a document about the Golden State Warriors. Sequence column indicates whether token is in description or document.

The architecture we employed for our experiments is presented in figure 4.

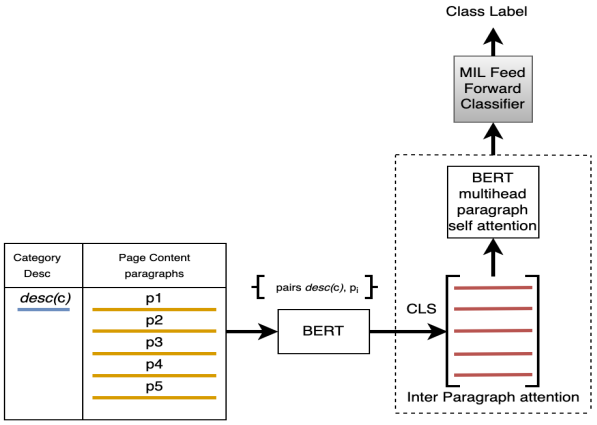


Fig. 4: Multiple instance learning model (MIL)

IV. Data and Experimental Results

A. Datasets

1) Wikipedia Dataset: The first dataset we used for our experiments is derived from Wikipedia; one of the strengths of our proposed models is that labeled training data can easily be generated automatically from publicly available data, without human labeling required.

Articles in Wikipedia are tagged with relevant categories. We use the first three sentences of the Wikipedia page corresponding to the category, if it exists, as the tag description. If no Wikipedia page for the category exists, it is omitted from training. For example, a page about the Albert Einstein might be tagged with the category physics so the first three sentences of the Wikipedia page about physics is used for <$desc$(physics)>.

Using the procedure outlined in the previous paragraph, for each page d we generated triplets (<$desc(c_i)$>,<d>,1)[1], where $i \in I$ and $|I|$ is the number of categories page d is tagged with that have a corresponding Wikipedia page. With the same procedure, we generated an equal number of triplets (<$desc(c_j)$>,<d>,0), where $j \in J$ with $|J| = |I|$ and $j \notin I$ using randomly selected categories different than those d was tagged with. For the experiments in section IV we used a partial dump of Simple English Wikipedia[2] with 4633 pages, 25458 paragraphs, and 6338 tags.

2) Proprietary Reference Manual Dataset: The second dataset we used consists of a corporate product reference manual (PRM), which contains documents about the products and services a US-based insurance company offers. The PRM is stored in a content management system (CMS), and organized using tags and a tree/subtree system that associates can access using a standard web interface. Each document was given relevant tags by a human expert at the time it was authored and placed in the CMS.

We note that in addition to running overnight as mentioned in section II-C, our model can be used at the time of authoring a document. Figure 5 shows an interface which suggests relevant predefined tags for a document based on its contents; authors may add additional new tags as well. The data collected through this process can further help refine and improve the model, especially instances where the author disagrees with the model's suggestion.

To generate descriptions for each tag, the authors used the internal corporate glossary and external web search to come up with a two-to-three sentence definition of each. Creating descriptions was manual, however, it took less than half a day; we suspect volumes and timelines will be similar with other companies and data sources. We then used a process identical to the one used with the Wikipedia dataset to create an equally balanced PRM dataset. There were 2055 pages, 40319 paragraphs and 42 tags. Note that pages of the PRM are less structured than Wikipedia, with many single-sentence paragraphs that provide relatively little context for paragraph-based modeling.

[1]For PM, MIL: (<$desc(c_i)$>,<p_k>,1) paragraphs $p_k \in d$

[2]https://dumps.wikimedia.org/simplewiki/

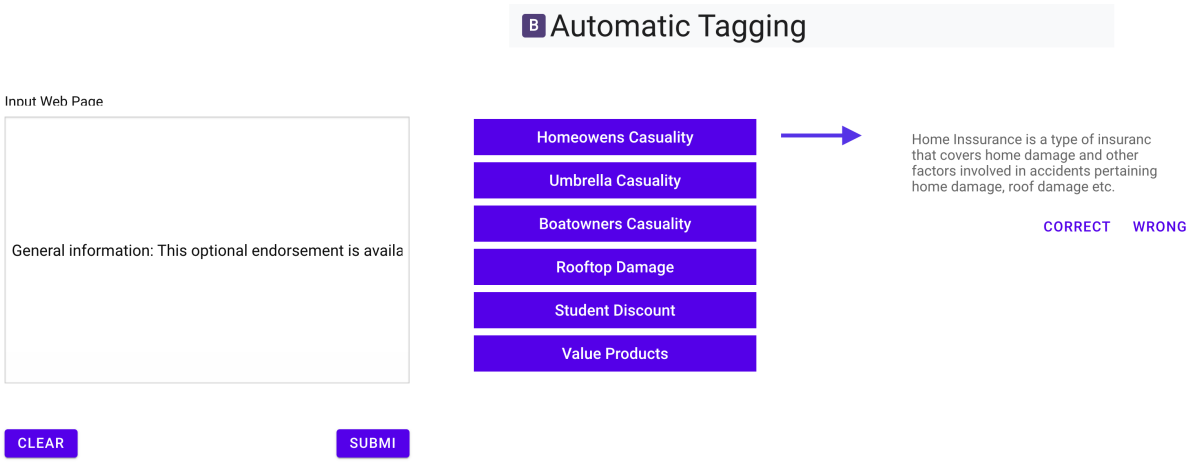


Fig. 5: Web Application for Automatic Tagging

B. Empirical results

Next we present three sets of experiments using models built in Python with Pytorch [8] and the Transformers library [9]. In section IV-B1 we compare our proposed construction to a multi-label classification approach to compare performance in the presence of long-tailed label distributions. Next, in IV-B2 we show results of our algorithm on unseen tags (zero-shot learning). To end, we show in section IV-B3 how transfer learning affects our zero-shot results.

1) Comparison to multi-label model (M-L): We benchmark our models against a BERT for sequence classification Multi-label model [9] for both the Wikipedia and PRM datasets using an (80%, 20%) train-test split of pages, (3706, 927) and (1644, 411) respectively. As $|C|$ increases, it becomes increasingly difficult for multi-label models to perform well and to appropriately measure their performance, especially given skewed tag distributions mentioned in I. With 6338 and 42 unique tags in the Wikipedia and PRM datasets respectively, these problems are pronounced, and our proposed models outperform the base model by large margins as shown in table II.

Given the difficulty of training a multi-label model on such so few examples, we also compare against a simple TF-IDF model. Using the category description as a query, we compute a TF-IDF score for each document in each training dataset. Documents with score greater than the mean score are labeled positive. We built the TF-IDF model with scikit-learn [10].

To measure performance in the multi-label setting, we use **percent true positives (PTP) (e.g. recall)** given by $PTP = Pred_{TP}/Actual_{TP}$ which captures the proportion of correctly predicted positive labels for each document. Metrics such as micro-averaging & macro-averaging, Hamming loss, and exact match ratio often overstate performance with highly sparse labels, as models tend to correctly classify a majority of true negatives (by "always predicting 0"). As seen in table IV our models have high accuracy in a balanced class binary classification setting, indicating an extremely low frequency of false positives, making PTP an appropriate metric on which to compare multi-label performance.

To further understand how the models perform as the number of tags per document increase, we break down the PTP by number of tags associated with documents in the dataset. For example, the PTP of all documents with two associated tags is in sub-column 2 of **PTP by Tag Count** in table II. We also average out the tag level average count positives and is represented in column **Avg PTP**. Our models all achieve an average PTP performance of greater than 92% on Wikipedia compared with a baseline 14% and greater than a PTP of 74% on the PRM dataset versus a baseline of 46%.

2) Binary Classifier Results (Zero-shot testing): To highlight the ability of our proposed architectures to handle zero-shot tags, we use an (80%, 20%) train-test split on both Wikipedia and PRM datasets at the tag level such that $(C_{train} \cap C_{test}) = \emptyset$, (5071,1267) and (33,9) respectively. Furthermore, we evaluate the performance of each model on the test portion of both datasets; **In-domain** results are for matching train-test datasets (e.g. both from Wikipedia) and **Out-domain** results are for differing train-test datasets (e.g. train from Wikipedia, test from PRM).

As a baseline, we compare our proposed models against the task-aware representation of sentences (TARS) model presented in [4]. Notably, we are able to test the performance of a pre-trained TARS model and our models trained on Wikipedia by performing link prediction on the PRM dataset, which is out-domain for both. Results are shown in table **??**.

Table IV shows that all our proposed models are successful on unseen in-domain tags and perform reasonably well even on unseen out-domain tags. We note part of the performance of our classifiers is likely due to the similarity between our task and next sentence prediction used in training BERT [6]. We highlight the potential for models with our proposed MIL architecture to achieve accuracy of 90% or 74% on domain-specific data with limited (PRM-trained) or *no* (Wikipedia-trained) labeled data respectively.

3) Transfer Learning Results: In order to test the effect of transfer learning, we pre-trained all our models on the Wikipedia dataset and fine-tuned them on the PRM. The derived model is thus expected to perform well on both the datasets. From table V we can see that the fine-tuned models have better scores on both datasets than the models trained on a single domain (table IV). All testing is done in the same way as in section IV-B2, so the reported metrics are an accurate measure of zero-shot performance. We highlight that the PM architecture with transfer learning achieves the highest overall accuracy of any experiment on our proprietary PRM dataset of 98%.

Dataset	Model	Avg PTP
Wikipedia	TF-IDF	0.29
	M-L	0.14
	DM	0.97
	PM	**0.99**
	MIL	0.92
PRM	TF-IDF	0.69
	M-L	0.46
	DM	0.76
	PM	0.74
	MIL	**0.85**

TABLE II: **Multi-label results:** Comparison of proposed architectures with traditional multi-label (M-L) approach.

Model	MAP@1	MAP@6	Recall	F1
HuggingFace	0.20	0.17	0.15	0.13
Flair TARS	0.05	0.15	0.21	0.20
Our DM	**0.58**	**0.60**	**0.98**	**0.72**

TABLE III: **Zero Shot Benchmark results:** Comparison of Mean Average Precision at K and F1 Sample metrics of our Zero Shot model compared to HuggingFace and Flair TARS model on the PRM dataset.

Dataset	Model	In-domain		Out-domain	
		ACC	AUC	ACC	AUC
Wikipedia	DM	0.98	0.98	0.69	0.74
	PM	**0.98**	**0.98**	0.62	0.65
	MIL	0.95	0.95	**0.74**	**0.74**
PRM	DM	0.76	0.80	**0.79**	**0.80**
	PM	0.74	0.73	0.76	0.78
	MIL	**0.90**	**0.90**	0.75	0.75

TABLE IV: **Binary classifier (Zero-shot) results:** Comparison of accuracy and AUC for proposed architectures. For model trained on Wikipedia, Out-domain is PRM and vice-versa.

C. Summary of results

Our experiments demonstrate the strength and flexibility of a semantics-based approach to tagging, regardless of architecture. The PM model outperforms all other variants for the large well-structured Wikipedia dataset. For unstructured datasets like the PRM, our MIL model has the highest performance as all sections of a page are pooled and attended to, helping to overcome the limitations of low-context paragraphs. Transfer learning shows excellent performance on proprietary data for application in industry.

Experiment	Model	PRM		Wikipedia	
		ACC	AUC	ACC	AUC
Wikipedia trained	DM	0.98	0.98	0.98	0.98
model fine-tuned	PM	**0.98**	**0.98**	**0.99**	**0.99**
on PRM	MIL	0.95	0.92	0.73	0.72

TABLE V: **Transfer learning results:** Comparison of accuracy and AUC for proposed architectures using transfer learning from Wikipedia to PRM.

V. Background and Related Work

Automatic tagging of text-based instances can be and is often treated as a multi-label classification problem [11] which have been explored in many contexts including, marketing [12], medical documents [13], and web pages [14].

Recently proposed Question-Answering models [15], [16] can answer questions given some text-based context. To do so, they predict locations of passages containing the answer to a given question from the context. These models learn appropriate embeddings for both the question and the context, where language structure and semantics are taken into account. Recent work has shown that attention-based architectures, more specifically transformers achieve SOTA performance for this task [6], [17]. Our construction is similar to the one used in QA models.

In recent years, Zero-shot learning (ZSL) has gained considerable attention. In this setup, a learner is expected to recognize testing examples from classes not previously seen when training [18].

[4] take a similar QA-like "task-aware" approach to text classification as our proposed model, and achieve excellent performance on zero-shot tasks making it a fitting baseline to compare against. As noted above, our model uses more context by including a description of the tag, and handles longer documents with the PM and MIL architectures. In addition we outline the construction of a larger training dataset based on Wikipedia, which can easily be expanded for even more robust training.

In [19] a similar idea is proposed by using text-based label definitions for the tagging problem zero-shot problem. However, the algorithm requires a class taxonomy (which don't always exist) and uses BERT only for text representation. In contrast, our construction is simpler and fundamentally based on the self-attention mechanism provided by transformer models.

In [20], the long-tailed class distributions problem often encountered in practice for multi-labeled problems is addressed. However, the approach is based on a new proposed loss function customized for computer vision problems.

VI. Conclusions and Future Work

Large transformer-based language models like BERT have greatly expanded the ways in which natural language processing can be applied to problems in industry. Using the flexibility and context awareness inherent in such models we developed a open-ended text tagging model that not only significantly outperforms standard multi-label approaches, but works for unseen tags as well. We demonstrated the power of our approach on two distinct datasets, and highlighted how using a publicly available dataset and transfer learning allow for highly-performant domain-specific models with limited labeling required, including the zero-shot case. Given the ubiquity of places where tagging text is relevant, we believe our approach can be successfully applied in a range of applications to improve accuracy and minimize model maintenance needs.

As future work we are interested in exploring this construction in the framework of machine teaching [21]. The goal of machine teaching is to design the optimal training data to drive the learning algorithm to a target model. In our case, the goal would be to explore human-in-the-loop approaches where the attention generated from the zero-shot model predictions could be used by the human to refine and improve the label definitions and in return the model could improve predictive performance.

References

[1] S. A. Fadaee and M. A. Haeri, "Multi-label classification using link prediction," 2020.

[2] M. Yuan, J. Xu, and Z. Li, "Long tail multi-label learning," in *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, 2019, pp. 28–31.

[3] T. Wei and Y.-F. Li, "Does tail label help for large-scale multi-label learning," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 7 2018, pp. 2847–2853. [Online]. Available: https://doi.org/10.24963/ijcai.2018/395

[4] K. Halder, A. Akbik, J. Krapac, and R. Vollgraf, "Task-aware representation of sentences for generic text classification," in *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 3202–3213. [Online]. Available: https://www.aclweb.org/anthology/2020.coling-main.285

[5] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," 2016.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[7] M. Ilse, J. M. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," *arXiv preprint arXiv:1802.04712*, 2018. [Online]. Available: https://arxiv.org/abs/1802.04712

[8] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[9] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[11] D. Xu, Y. Shi, I. W. Tsang, Y. S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2409–2429, 2020.

[12] J. Salminen, V. Yoganathan, J. Corporan, B. J. Jansen, and S.-G. Jung, "Machine learning approach to auto-tagging online content for content marketing efficiency: A comparative analysis between methods and content type," *Journal of Business Research*, vol. 101, pp. 203 – 217, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0148296319302607

[13] D. Pan, X. Zheng, W. Liu, M. Li, M. Ma, Y. Zhou, L. Yang, and P. Wang, "Multi-label classification for clinical text with feature-level attention," *2020 IEEE 6th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*, pp. 186–191, 2020.

[14] R. You, S. Dai, Z. Zhang, H. Mamitsuka, and S. Zhu, "Attentionxml: Extreme multi-label text classification with multi-label attention based

recurrent neural networks," *CoRR*, vol. abs/1811.01727, 2018. [Online]. Available: http://arxiv.org/abs/1811.01727

[15] F. Zhu, W. Lei, C. Wang, J. Zheng, S. Poria, and T.-S. Chua, "Retrieving and reading: A comprehensive survey on open-domain question answering," 2021.

[16] Z. Abbasiantaeb and S. Momtazi, "Text-based question answering from information retrieval and deep neural network perspectives: A survey," 2020.

[17] B. van Aken, B. Winter, A. Löser, and F. A. Gers, "How does bert answer questions?" *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, Nov 2019. [Online]. Available: http://dx.doi.org/10.1145/3357384.3358028

[18] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, Jan. 2019. [Online]. Available: https://doi.org/10.1145/3293318

[19] J. Lu, L. Du, M. Liu, and J. Dipnall, "Multi-label few/zero-shot learning with knowledge aggregated from multiple label graphs," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 2935–2943. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-main.235

[20] T. Wu, Q. Huang, Z. Liu, Y. Wang, and D. Lin, "Distribution-balanced loss for multi-label classification in long-tailed datasets," 2020.

[21] X. Zhu, A. Singla, S. Zilles, and A. N. Rafferty, "An overview of machine teaching," 2018.