Evaluating Neural Multi-Field Document Representations for Patent Classification

Subhash Chandra Pujari[1,2], Fryderyk Mantiuk[1,3], Mark Giereth[4], Jannik Strötgen[1] and Annemarie Friedrich[1]

[1] *Bosch Center for Artificial Intelligence, Renningen, Germany*
[2] *Institute of Computer Science, Heidelberg University, Heidelberg, Germany*
[3] *Duale Hochschule Baden-Württemberg, Stuttgart, Germany*
[4] *Robert Bosch GmbH, Stuttgart, Germany*

Abstract

Patent classification constitutes a long-tailed hierarchical learning problem. Prior work has demonstrated the efficacy of neural representations based on pre-trained transformers, however, due to the limited input size of these models, using only title and abstract of patents as input. Patent documents consist of several textual fields, some of which are quite long. We show that a baseline using simple tf.idf-based methods can easily leverage this additional information. We propose a new architecture combining the neural transformer-based representations of the various fields into a meta-embedding, which we demonstrate to outperform the tf.idf-based counterparts especially on less frequent classes. Using a relatively simple architecture, we outperform the previous state of the art on CPC classification by a margin of 1.2 macro-avg. F1 and 2.6 micro-avg. F1. We identify the textual field giving a "brief-summary" of the patent as most informative with regard to CPC classification, which points to interesting future directions of research on less computation-intensive models, e.g., by summarizing long documents before neural classification.

Keywords

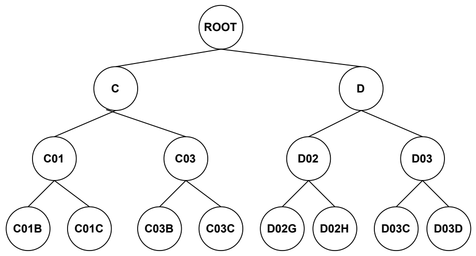
patent classification, long-tailed classification, neural document representations

1. Introduction

Organizations must be aware of competitors' Intellectual Property (IP) artifacts as IP litigation might involve a considerable loss in revenue or delay product launch. Carving out a competitor's IP portfolio or the relevant IP landscape often relies on classifying patents by their content. For example, the Cooperative Patent Classification[1] (CPC) is a large taxonomy used by major patent offices for their internal organization. Once submitted to a patent office, each patent application is manually labeled with a set of labels taken this taxonomy. Within companies, a second step in managing or understanding patent portfolios usually consists in organizing patents into internal or use-case specific classification schemes. Besides the obvious use case of automating

BIR 2022: 12th International Workshop on Bibliometric-enhanced Information Retrieval at ECIR 2022, April 10, 2022.
subhashchandra.pujari@de.bosch.com (S. C. Pujari); fryderyk.mantiuk@de.bosch.com (F. Mantiuk); mark.giereth@de.bosch.com (M. Giereth); jannik.stroetgen@de.bosch.com (J. Strötgen); annemarie.friedrich@de.bosch.com (A. Friedrich)

[1] https://www.cooperativepatentclassification.org



Level	ID	Description
Section	C	Chemistry; Metallurgy
Class	C01	Inorganic Chemistry
Subclass	C01C	Ammonia; Cyanogen;

Figure 1: An excerpt from the Cooperative Patent Classification (CPC) taxonomy.

the first step, CPC classification is a good testbed for patent classification in general due to the free availability of labeled data, and has been the target of research for decades already [1, 2, 3, 4, 5, 6, 7, 8, 9]. The CPC taxonomy arranges labels into up to nine hierarchical levels. At the fifth level, the taxonomy has about 240k labels with a very skewed distribution with a very long tail. Following previous works [7, 8, 9], we restrict our evaluation to the first three levels of hierarchy – as exemplarily shown in Figure 1 – which still results in a very challenging long-tailed hierarchical classification task with 767 labels.

Patents are structured text documents with multiple fields, e.g., title, abstract, description, and claims. Among these, only the claims, which have to be interpreted in context of the description, are legally binding. The remaining fields are often drafted less carefully, or even intentionally conceal a patent's content. In earlier research on *non-neural* text classification [2, 3, 4, 5, 10], document representations for long documents were obtained for instance using tf.idf-based methods. Despite being robust, they ignore sequence information and do not, in contrast to more recent neural language models, profit from self-supervised pre-training. Due to the limited input size of transformers such as BERT [11], which can process only input up to 512 word-piece tokens corresponding to only few sentences, prior research on *neural* patent classification [8, 9, 12] has mainly relied on title and abstract only. The latter is obviously problematic as these fields are often rather broad, and hence often do not precisely describe the patent's content. We are also not aware of a systematic study comparing the various possible input fields for patent classification to exploit potentially complementary information across fields.[2] In sum, prior research either did not leverage the strength of transformers, or uses only title and abstract to compute a document representation. In this work, we hence propose a neural system architecture with a pre-trained transformer-based neural language model [14] as backbone, but incorporating embeddings derived from various textual fields.

As a first step, we enrich the USPTO-70k dataset, which originally only contained `title` and `abstract`, with the four additional patent fields `claims`, detailed description (`detail-desc`), `brief-summary`, and figure description (`fig-desc`). The latter three fields are sub-fields of description within USPTO patents. Based on this enriched dataset, we perform various experiments systematically comparing the performance of a non-neural hierarchical classifier [15], whose representations are based on tf.idf and that uses SVMs internally, with an extension of a state-of-the-art neural model [9]. We evaluate the various text field embeddings in terms

[2]Ingwersen [13] introduced the concept of polyrepresentation in the field of cognitive IR theory, which states that the uncertainty of an information retrieval system decreases by incorporating multiple representations of full-text semantic entities, e.g., sentence, paragraph, sections, etc., into a document representation.

of their informativeness with respect to the task of classifying patents according to the CPC taxonomy, finding that models using the `brief-summary` are very effective. We also find that information from the fields is complementary, i.e., the models using all input fields outperform their counterparts using subsets thereof. This finding holds both in the case of non-neural and neural systems. Finally, when combining several embeddings into a meta-embedding, we found vector summation to outperform concatenation. Our further analysis shows that using additional textual information works best especially for the difficult infrequent labels, i.e., in few-shot scenarios.

In summary, we propose a novel neural system for patent classification, demonstrating how and that the various textual sub-fields of patents can be an effective source of information. In particular, our contributions are as follows:

- We enrich the USPTO-70k dataset of [9], which contains only titles and abstracts, with four additional patent fields, making it as well as code available to foster future research.[3]
- We propose an approach to efficiently generate an effective document representation using a transformer-based model incorporating complementary information from multiple textual patent fields. We demonstrate that the additional information increases classification performance by a considerable margin.
- Unlike previous works, e.g., [8, 9], we also evaluate the model performance in a few-shot setting, where our proposed approach fairs particularly well in the least frequent label group (showing 44% better F1-macro).

2. Related Work

Approaches to patent classification differ across several dimensions: there are neural and non-neural methods, approaches exploiting the full document text vs. ones relying on title and abstract only, and techniques performing hierarchical classification vs. those tackling the coarsest (CPC) class granularity only.

Full-text-based approaches. With the underlying bag-of-word representation, a tf.idf feature vector incorporates the complete document text into a document representation. Fall et al. [2] experiment with title, abstract, claims, description, and the meta-data fields and find that using the first 300 words of the title, inventors, applicant, abstracts, and description works best. Several works [3, 4, 5] show that the description is more informative than other sections of a patent, in particular, the initial part of the description. Benites et al. [10] rank first in the ALTA 2018 Shared Task on patent classification [16] and use the full text of the patent documents. However, their method predicts only nine labels at the Section level, while we address hierarchical patent classification with 767 labels across three granularities.

Neural Models. Analyzing the submissions of the TREC deep learning 2019 track, Carswell et al. [17] conclude that BERT-based methods substantially outperform earlier used text representation techniques. Based on this finding, Lin et al. [18] divide the timeline of deep learning models into pre-BERT and BERT eras. The two primary methods for feature vector generation in the pre-BERT era are Convolutional Neural Networks (CNNs) and Recurrent

[3]https://github.com/boschresearch/multifield_patent_classification_bir2022

Neural Networks (RNNs), including its successors, i.e., GNNs and LSTMs. DeepPatent by Li et al. [7] applies a CNN on top of word embeddings for the first 100 words from title and abstract. They compare the results with a non-neural baseline, which uses a tf.idf vector generated with the complete document text. However, the insights of this comparison are limited as only the micro-precision value outperformed the non-neural model. Grawe et al. [19] apply an LSTM on top of word embeddings and compare using the first 150 words of the description against using the first 400 words. Risch and Krestel [20] pre-train fastText word embeddings on a large corpus of patent documents and combine the word embeddings using Gated Recurrent Units (GRUs). They also use only title and abstract, but they include the first 300 words. Lee and Hsiang [8] is the first one to apply BERT for the patent classification task. They report that using only the first claim gives results at par with title and abstract. Unlike [8], Pujari et al. [9] consider patent classification as a hierarchical multi-label classification problem and propose a hierarchical transformer-based multi-task model that trains an intermediate SciBERT layer with title and abstract as input text. Comparing BERT and SciBERT on a patent classification task, Althammer et al. [21] found that the SciBERT model performs better. The transformer-based language model is computationally expensive, therefore, subject to a maximum sequence length constraint, accommodating only 512 word-piece tokens.

Transformers with longer inputs. Long-text transformer models reduce the computational overhead with the sparse attention mechanism, increasing the sequence length to 4096 tokens. Zaheer et al. report state-of-the-art patent classification results using Big Bird [12] using title, abstract and claims. Although effective, the transformer-based model is still incapable of accommodating the complete document text into a document representation. Therefore, in this work, we look into text pruning techniques for a more informative document representation.

Few-shot/long-tailed text classification. Besides an effective document representation for a long multi-field document, a major challenge with CPC classification is the prediction of less frequent labels. Mullenbach et al. [22] proposed a method for few-shot learning with attention-based weights for combining the label-based embedding and incorporating it into a document representation. Further, Rios et al. [23] extended this method [22] by incorporating information from a hierarchical taxonomy using a GCNN [24]. Surveying the literature proposing novel datasets [25, 26, 27] for a few-shot learning setting, we do not find a clear definition of the criterion used for categorizing a label into a few-shot category. Therefore, we create frequency-based label groups for evaluating a few-shot setting.

3. Model

Our classification model architecture is similar to that proposed in [9]. However, instead of using only the concatenated text of title and abstract as input to the transformer, we compute embeddings of various patent fields, aggregating them into a meta-embedding using vector summation or concatenation. **Field Embeddings.** We use PatSciBERT, i.e., the SciBERT model [14] finetuned on CPC classification by [9], to generate embeddings for the first 510 word-piece tokens of each textual field. This is motivated by the findings of previous works that the initial part of a field's text is often more informative than the remainder [3, 4, 5]. In most experiments,

Table 1
Abbreviations.

Symbols	Description
t	title
a	abstract
c	claims
d	detail-desc
bs	brief-summary
fd	fig-desc
$+$	text concatenation
$e(\cdot)$	A method which tokenizes an input text and calculates the sequence embedding using a transformer-based LM.
$e_f(\cdot)$	Same as $e(\cdot)$, however, the LM is finetuned during the task.
;	vector concatenation
$\oplus$	vector summation

we do not fine-tune PatSciBERT. For word-piece tokenization, we use a BertTokenizer[4] initialized with Sci-BERT's vocabulary. The 768-dimensional last hidden state of the [CLS] token is used as the text field's embedding. We denote this embedding as $e(\cdot)$ when not fine-tuning; $e_f(\cdot)$ is used when fine-tuning the underlying language model.

Aggregation. For aggregating several field embeddings into a document representation x_i (for the ith instance), we experiment with two simple aggregation methods, i.e., vector summation ($\oplus$) and concatenation (;).

Classification Model. We use a Transformer-based Hierarchical Multi-task Model (THMM) architecture similar to [9], with x_i as input, and one classification head per label. The heads consist of three-layer perceptrons with a binary softmax head, predicting whether the label applies to the document or not. The hierarchical taxonomy links define the input to the classification heads as the concatenation of the document representation x_i and the output of the second hidden state of the respective parent heads.

4. Experimental Setup

In this section, we describe our experimental setup. First, in Section 4.1, we describe the steps taken for enriching the USPTO-70k dataset by incorporating additional fields taken from *USPTO bulk download*.[5] Section 4.2 introduces the evaluation metrics for hierarchical multi-label classification, which are used for the analysis of results in Section 5. Also, here we define the label groups based on taxonomy level, label frequency, and section information. We compare our results with the neural and non-neural baselines as described in Section 4.3, whereas our experimental setup is defined in Section 4.4.

[4]https://huggingface.co/transformers/v2.4.0/model_doc/bert.html#berttokenizer
[5]https://patentsview.org/download/data-download-tables

Table 2
In practice, not all patents contain all textual fields. We here show, for the case of the USPTO-70k dataset, the number of instances which do not contain particular subfields of `brief-summary`.

field-name	train-set	dev-set	test-set
total number of instances	50,625	10,000	10,000
invention summary missing	5,836	1,428	1,413
background of invention missing	3,888	845	855
technical-field missing	28,304	5,111	5,008

4.1. Dataset

We use the USPTO-70k dataset released by Pujari et al. [9], containing 50k train, 10k test and 10k dev instances. The instances are labeled with 9, 128, 630 unique labels from each of the first three CPC[6] levels, i.e., section, class, subclass. The original USPTO-70k dataset provides only `title` and `abstract`. We enrich the dataset with additional patent fields and make the enriched dataset publicly available to ease future work. A patent document contains the four main text fields `title`, `abstract`, `claims`, and `description`, with the latter being the longest and most detailed. The USPTO dump aggregates the subfields within the `description` into three groups: `brief- summary`, `fig-desc`, and `detail-desc`. With approximately 1.8k word-piece tokens, `brief-summary` is very concise in contrast to the elaborate `detail-desc` field with approximately 9.5k tokens (see Table 2). As shown in Table 2, `brief-summary` often contains several subfields, however, as there is no strict structure or completeness required, we simply use the concatenation of these texts.

4.2. Evaluation

Metrics. In line with [9], we evaluate the models using hierarchical precision, recall and F1-score as proposed by [28], defining precision as $hP = \frac{\sum P_i \cap T_i}{\sum P_i}$ and recall as $hR = \frac{\sum P_i \cap T_i}{\sum T_i}$. For each test instance i, the predicted label set P_i consists of all predicted labels with their ancestors. Similarly, the true label set T_i contains true labels, including ancestors. Since within a CPC taxonomy, each child label has a single ancestor, we add missing ancestors for each predicted label. Most previous works [7, 8, 12] evaluate the model performance using the micro-average scores, which do not adequately reveal the performance of a model for the less frequent labels. Therefore, we compute macro-scores for each of the three measures as an average of scores obtained across labels due to the skewed label distribution. For example, the macro-F1 score is computed as an average F1-score for each label. Also, we segregate the labels into groups according to various criteria for more fine-grained analysis. The macro-F1 for a group is computed as the average of the per-label F1 scores of the labels in the respective group. The grouping strategy is defined with three criteria: frequency, level, and section information. Table 3 shows details of groups within each of the three settings, which are further explained below.

[6]CPC taken at 2020.06.

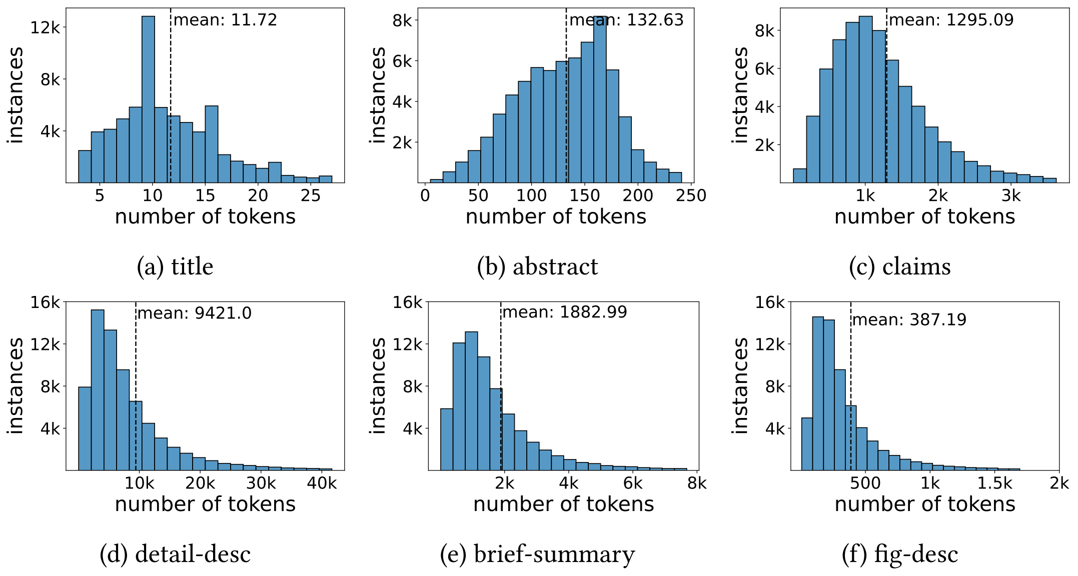


Figure 2: Word-piece token distribution for different patent fields.

Label Grouping. For our analyses, we perform several groupings of labels in order to compute (macro-average) results by label.

Grouping by Label Frequency. The less frequent labels capture the fine-grained information and thus are often more informative than the more frequent ones. Previous works evaluate the performance of a model for these less frequent labels under a few-shot setting. However, there is no standard threshold for what constitutes a minority class in few-shot text classification.[7] Therefore, instead of sticking to a single value as a measure of the few-shot category, we define four frequency-based label groups with a label frequency threshold of 10, 50, and 100, respectively. Table 3 shows the number of labels within each label group.

Grouping by Level. We use the hierarchical taxonomy information for dividing the labels into groups based on the hierarchical level. Since the USPTO dataset consists of labels from the first three levels of the CPC taxonomy, we create three label groups with 9, 128, 630 labels each.

Grouping by Section (Domain). In this setting, we align a section label and all its child nodes as a single group, creating nine groups in total, essentially performing a topical grouping as shown in Table 3. Group B contains the largest number of labels followed by F. The group Y consists of only 15 labels used as a general tagging labels for new technologies or a general tagging label for cross-sectional technologies spanning over several sections.

4.3. Baselines

TwistBytes (TB). As a *non-neural* baseline, we use TwistBytes [15], a Local Classifier per Node (LCN) approach which trains a Support Vector Machine (SVM) model [29] as a base classifier for

[7]For example, MIMIC-III [26], EURLEX57K [25], AMAZON13K [27] have few-shot categorization thresholds of 5, 50 and 100 respectively.

Table 3
Shows the distribution of labels within three label groups.

Grouping Criterion	Grouping Identifier	Definition	Count (subclass) total=630	# Instances
	1-10	$f_i \leq 10$	114	1,085
Label	11-50	$10 < f_i \leq 50$	238	5,079
Frequency	51-100	$50 < f_i \leq 100$	91	5,897
	100+	$100 < f_i$	187	47,486
			Count (all labels) total=767	
	1	Level 1 CPC taxonomy	9	50,625
Level	2	Level 2 CPC taxonomy	128	50,625
	3	Level 3 CPC taxonomy	630	50,625
			Count (all labels) total=767	
	A	Human Necessities	100	8,157
	B	Performing Operations, Transporting	199	18,167
	C	Chemistry, Metallurgy	103	10,474
	D	Textiles; Paper	44	6,791
Section	E	Fixed Constructions	38	2,390
	F	Mechanical Engineering; Lightning; Heating; Weapons; Blasting	119	15,591
	G	Physics	93	20,458
	H	Electricity	56	20,206
	Y	General Tagging	15	9,360

each label in the class hierarchy. TwistBytes uses the sibling strategy when training a classifier, i.e., using the subset of the training data consisting of the instances with the label addressed by the classifier or those of the respective siblings. The tree is traversed from the root node to leaf nodes during prediction, predicting a label at each hop using the label-specific classifier. A child label classifier is traversed if the probability of the parent label is more than a user-defined threshold. We use the same parameter values as defined by [15] using a decision function threshold value of -0.25 and the TF-IDF vector of dimension 70k.

THMM with $e_f(t{+}a)$. As a *neural* baseline, we chose the THMM setting of [9] (see Section 3). They generate a document representation, concatenating the title and abstract and passing it through the SciBERT model. The SciBERT model weights are finetuned during training.

4.4. Experimental Setup and Hyperparameters

To make the results comparable, all models use the same hyperparameters, which are based on the best values found by [9]. The hidden layer size of all dense layers in the classification heads is 256, dropout is set to 0.25 across layers and we use a batch size of 64. Contrary to Pujari et al., we use a learning rate of 10^{-3}, because we are not fine-tuning the language model. We train all models for a maximum of 50 epochs with early stopping if the F1-macro for the dev dataset does not increase for 7 epochs. As indicated in Figure 2, the majority of the sections exceed the 512-token limit imposed by SciBERT and many of the other Transformer-based models. For all of the sections where this is the case, we set the maximum input length to 512 tokens. In sections that can instead fit in smaller sequences, we reduce the maximum input length of the tokenizer for efficiency reasons. For `title` we use a maximum length of 64, and for `abstract`, a maximum length of 256. For all implementations, we use Python with the libraries *TensorFlow*[8] and *Keras*[9] [30]. For integrating SciBERT, we make use of the *transformers* [31] library from *Huggingface*[10].

5. Experimental Results

Our experiments aim to identify the most informative text field embeddings and best aggregation method for combining them into a document representation. Section 5.1 compares our best performing model to the neural and non-neural baselines, including an analysis by label frequency. In Section 5.2, we compare the performance when using various combinations of fields as input broken down by label frequency, domain, and hierarchy level.

5.1. Performance of Models Using Various Document Representations

Informativeness of Individual Field Embeddings. In order to assess the contribution of different fields towards the informativeness w.r.t. the document classification task, we use one field at a time, generate a document representation, and evaluate it on the CPC classification task. As we can see in Table 4, the `brief-summary` is the most informative field in terms of overall performance, showing a high score across metrics and clearly outperforming models leveraging `abstract` and `claims`. Unlike `detail-desc`, a `brief-summary` often includes the invention summary and precise details on the technical field of an invention, which might be a possible reason for its higher informativeness compared to other patent fields. As the `title` field contains a few terms describing an invention in absolute brevity, a document representation based on `title` can identify some labels with high precision, but only has a low recall. Similar conclusions might be drawn for the `fig-desc`, as it contains particular domain-related terms, and for legal implications claims, which are very specific to an invention. In contrast, no such legal boundation holds for `abstract`, thus it is often drafted in a broad and imprecise manner, partially explaining its limitations for use in classification tasks such as ours. Thus, it shows a high recall score but lower precision.

Aggregating Information from Several Fields. Next, we combine information from several field embeddings (see Table 4). First, comparing vector summation and concatenation, we see a clear advantage when using summation ($\oplus$) over concatenations (indicated by ;) as aggregation method. Second, we observe that in both cases, adding more information helps.

[8]https://www.tensorflow.org/
[9]https://keras.io/
[10]https://huggingface.co/transformers/

Table 4
Model performance for different document representation inputs.

		macro-avg.			micro-avg.		
Model	doc-rep	P	R	F1	P	R	F1
TB	$tf\text{-}idf(t+a)$	39.8	19.1	24.2	65.1	53.4	58.7
TB	$tf\text{-}idf(t+a+c)$	43.4	20.6	25.9	66.9	56.1	61.0
TB	$tf\text{-}idf(t+a+c+d)$	43.4	21.3	26.5	68.8	59.3	63.7
TB	$tf\text{-}idf(t+a+c+d+bs)$	45.4	23.4	28.8	69.9	60.8	65.0
TB	$tf\text{-}idf(t+a+c+d+bs+fd)$	45.5	23.6	28.9	69.9	60.9	65.1
THMM	$e_f(t+a)$	42.6	**36.7**	37.7	66.6	63.3	64.9
THMM	$e(t)$	35.6	19.5	23.6	62.8	49.0	55.0
THMM	$e(a)$	39.6	30.0	32.6	66.1	59.8	62.8
THMM	$e(c)$	41.1	27.5	31.0	67.5	58.0	62.4
THMM	$e(d)$	42.5	25.1	29.9	66.9	54.1	59.8
THMM	$e(bs)$	47.8	32.3	36.6	70.4	59.9	64.7
THMM	$e(fd)$	38.9	20.1	25.0	65.7	49.3	56.3
THMM	$e(t); e(a)$	40.9	30.7	33.4	67.6	60.4	63.8
THMM	$e(t); e(a); e(c)$	42.4	30.6	33.8	69.4	60.3	64.5
THMM	$e(t); e(a); e(c); e(d)$	43.9	31.0	34.6	68.7	61.4	64.8
THMM	$e(t); e(a); e(c); e(d); e(bs)$	45.8	31.4	35.5	70.0	62.1	65.8
THMM	$e(t); e(a); e(c); e(d); e(bs); e(fd)$	46.6	32.0	36.1	69.9	62.3	65.9
THMM	$e(t) \oplus e(a)$	40.9	30.2	32.6	66.6	60.1	63.2
THMM	$e(t) \oplus e(a) \oplus e(c)$	42.0	31.9	34.6	67.4	62.3	64.8
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d)$	43.9	33.3	35.8	68.4	63.5	65.9
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(bs)$	47.7	34.6	38.3	70.4	64.0	67.1
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(bs) \oplus e(fd)$	**48.6**	35.0	**38.9**	**70.2**	**65.0**	**67.5**

The F1-macro score being the primary metrics of our analysis, we see a significant gain in the score when adding the `brief-summary` to the document representation. Here, we observe that the performance across the neural and non-neural approaches is consistent. It means the informativeness of a field is independent of the text representation method.

Comparison of Best Configuration to Baseline Systems. First, we note that TwistBytes using tf.idf-based representations of the complete document text has a higher micro-F1 score than the THMM [9] with $e_f(t+a)$) as proposed by Pujari et al. [9]. This clearly demonstrates that there is relevant information in the additional text, and motivates us to combine multi-field embedding into a single *neural* document representation using an effective aggregation technique. Table 4 shows the evaluation results of our proposed approach (THMM with sum-based aggregation of six content fields, last row) compared to the baselines (TwistBytes and THMM with the concatenated `title` and `abstract` text as input). Our proposed approach outperforms both baselines in terms of macro-F1 and micro-F1 scores. When comparing our approach to the version of THMM proposed by [9], we see much improvement in precision with a slight dip in recall. Our proposed approach performs better across all three micro-average scores, i.e., precision, recall, F1.

Table 5
Macro scores for the best performing model and baselines over frequency-based groups.

			macro-avg.		
model	doc-rep	group	P	R	F1
TB	$tf\text{-}idf(t + a + c + d + bs + fd)$	1-10	1.80	00.7	0.90
TB	$tf\text{-}idf(t + a + c + d + bs + fd)$	11-50	31.7	11.7	16.0
TB	$tf\text{-}idf(t + a + c + d + bs + fd)$	51-100	**63.6**	23.4	32.0
TB	$tf\text{-}idf(t + a + c + d + bs + fd)$	100+	**64.6**	40.8	47.8
THMM	$e_f(t + a)$	1-10	11.1	7.1	7.5
THMM	$e_f(t + a)$	11-50	31.4	25.9	26.2
THMM	$e_f(t + a)$	51-100	45.9	**39.7**	40.9
THMM	$e_f(t + a)$	100+	56.8	**48.5**	51.4
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(b) \oplus e(f)$	1-10	**13.1**	**10.7**	**10.8**
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(b) \oplus e(f)$	11-50	**43.5**	**27.1**	**31.0**
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(b) \oplus e(f)$	51-100	57.4	37.0	**43.2**
THMM	$e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(b) \oplus e(f)$	100+	64.2	46.3	**52.5**

Performance by Label Frequency. Table 5 shows the results of our best performing approach (THMM with $e(t) \oplus e(a) \oplus e(c) \oplus e(d) \oplus e(b) \oplus e(f)$) and baselines for different frequency-based groups. We find that our approach works better overall, in particular, for the most infrequent label set, i.e., consisting of labels having a count less than 10, the macro-F1 is 44% better than the THMM with $e_f(t + a)$. The performance of the non-neural TwistBytes model is strong for more frequent labels, but poor for less frequent labels.

Summary. In our experiments, we identify `brief-summary` as the most informative patent field and vector summation as the most effective aggregation technique. Neural models outperform their non-neural counterparts especially in the case of infrequent labels.

5.2. Analysis of Field Embeddings by CPC Level, Label Frequency, and Domain

We here report a fine-grained analysis of model performance for three label groupings, focusing on macro-F1. Overall, models using the `brief-summary` field perform better than using all other fields across all label-grouping settings (see Figure 3), and excel in few-shot scenarios.

Level Hierarchy. On analyzing the level group results in Figure 3, we observe that at level 1, the performance is quite similar across fields, including `title` and `fig-desc`. However, for more fine-grained labels, especially at level 3, the `brief-summary` is more informative than another field embedding.

Frequency-based Groups. For the high-frequency label group, we see a similar performance for `abstract`, `claims`, and `brief-summary`. However, for labels with fewer instances, the performance for `brief-summary` is noticeably better.

Section/Domain. Using `brief-summary` consistently leads to better results across different sections, followed by the `abstract` (see Figure 3). However, in the case of D, H, Y as label groups, the relative gain in performance with `brief-summary` is marginal compared to `abstract`.

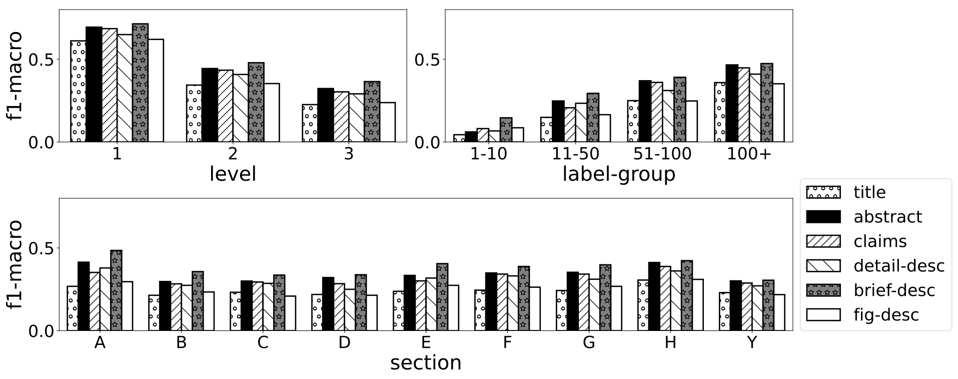


Figure 3: F1 macro distribution across three label group setting. The details for each of the label group are shown in Table 3.

6. Conclusion and Outlook

In this paper, we have addressed the challenge of classifying patents, which are long multi-field text documents. We have shown that performance of both non-neural and neural models benefits from leveraging a larger document context by combining text snippets from the various fields. As a first step, we have enriched the USPTO-70k patent dataset with four additional textual patent fields. Among these, we identify `brief-summary` as the most informative patent field in terms of overall performance and as being highly effective for classifying infrequent cases. Our experiments identify vector summation to perform better than concatenation.

While our model is conceptually simple and clearly outperforms previous work, it also points towards interesting directions for future work. A first step is clearly to try more advanced methods for creating meta-embeddings, such as incorporating adversarial techniques as in [32]. Second, another promising direction for patent classification is to employ variants of transformer-based neural language models (such as LongFormer [33] or Big Bird [12]) that incorporate larger text documents. In particular, as we have found the `brief-summary` field to be a very effective source of information, a potential future system could first apply a summarization method on the entire patent and then compute an embedding using such an extended language model. Finally, patents do not just consist of textual fields, but also include images or diagrams as well as further meta-data that will likely contain relevant information. The integration of these various types of information, e.g., in multi-modal approaches, certainly is another fruitful direction for research on patent classification.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. We also thank Tim Tarsi for fruitful discussions.

References

[1] H. Smith, Automation of Patent Classification, World Patent Information 24 (2002) 269–271. doi:10.1016/S0172-2190(02)00067-4.

[2] C. J. Fall, A. Törcsvári, K. Benzineb, G. Karetka, Automated Categorization in the International Patent Classification, SIGIR Forum 37 (2003) 10–25. doi:10.1145/945546.945547.

[3] J. Guyot, K. Benzineb, G. Falquet, myClass: A Mature Tool for Patent Classification, in: Proceedings of the International Conference of the Cross-Language Evaluation Forum (CLEF'10), CEUR-WS.org, Padua, Italy, 2010. URL: http://ceur-ws.org/Vol-1176/CLEF2010wn-CLEF-IP-GuyotEt2010.pdf.

[4] C.-H. Wu, Y. Ken, T. Huang, Patent Classification System Using a New Hybrid Genetic Algorithm Support Vector Machine, Applied Soft Computing 10 (2010) 1164–1177. doi:10.1016/j.asoc.2009.11.033.

[5] S. Verberne, E. D'hondt, Patent Classification Experiments with the Linguistic Classification System LCS in CLEF-IP 2011, in: Proceedings of the International Conference of the Cross-LanguageEvaluation Forum (CLEF'11), CEUR-WS.org, Amsterdam, The Netherlands, 2011. URL: http://ceur-ws.org/Vol-1177/CLEF2011wn-CLEF-IP-VerberneEt2011.pdf.

[6] K. Benzineb, J. Guyot, Automated Patent Classification, Current Challenges in Patent Information Retrieval (2011) 239–261. doi:10.1007/978-3-642-19231-9_12.

[7] S. Li, J. Hu, Y. Cui, J. Hu, DeepPatent: Patent Classification with Convolutional Neural Networks and Word Embedding, Scientometrics 117 (2018) 721–744. doi:10.1007/s11192-018-2905-5.

[8] J.-S. Lee, J. Hsiang, PatentBERT: Patent Classification with Fine-Tuning a pre-trained BERT Model, 2019. arXiv:1906.02124.

[9] S. C. Pujari, A. Friedrich, J. Strötgen, A Multi-Task Approach to Neural Multi-Label Hierarchical Patent Classification using Transformers, in: Proceedings of the 43rd European Conference on Information Retrieval (ECIR'21), Online, 2021. doi:10.1007/978-3-030-72113-8_34.

[10] F. Benites, S. Malmasi, M. Zampieri, Classifying Patent Applications with Ensemble Methods, in: Proceedings of the 16th Annual Workshop of The Australasian Language Technology Association (ALTA'18), Dunedin, New Zealand, 2018. URL: https://aclanthology.org/U18-1012.

[11] J. Devlin, M.-W. Chang, K. Lee, 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 (NAACL'19), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[12] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang, A. Ahmed, Big Bird: Transformers for Longer Sequences, in: Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS'20), online, 2020. URL: https://proceedings.neurips.cc/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

[13] P. Ingwersen, Polyrepresentation of information needs and semantic entities: Elements of a cognitive theory for information retrieval interaction, in: Proceedings of the 17th

Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94), Springer-Verlag, Berlin, Heidelberg, 1994, p. 101–110.

[14] I. Beltagy, K. Lo, A. Cohan, SciBERT: A Pretrained Language Model for Scientific Text, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP'19), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 3615–3620. doi:10.18653/v1/D19-1371.

[15] F. Benites, TwistBytes – Hierarchical Classification at GermEval 2019: Walking the Fine Line (of Recall and Precision), in: Proceedings of KONVENS'19, German Society for Computational Linguistics & Language Technology, Erlangen-Nürnberg, Germany, 2019, pp. 326–335. URL: https://konvens.org/proceedings/2019/papers/germeval/Germeval_Task1_paper_6.pdf.

[16] D. Mollá, D. Seneviratne, Overview of the 2018 ALTA Shared Task: Classifying Patent Applications, in: Proceedings of the 16th Annual Workshop of The Australasian Language Technology Association (ALTA'18), Dunedin, New Zealand, 2018, pp. 84–88. URL: https://aclanthology.org/U18-1011.

[17] N. Craswell, B. Mitra, E. Yilmaz, D. Campos, E. M. Voorhees, Overview of the TREC 2019 Deep Learning Track, 2020. arXiv:2003.07820.

[18] J. Lin, R. Nogueira, A. Yates, Pretrained Transformers for Text Ranking: BERT and Beyond, Morgan & Claypool Publishers, 2021. doi:10.2200/S01123ED1V01Y202108HLT053.

[19] M. F. Grawe, C. A. Martins, A. G. Bonfante, Automated Patent Classification Using Word Embedding, in: Proceedings of the 16th IEEE International Conference on Machine Learning and Applications (ICMLA'17), IEEE, Cancun, Mexico, 2017, pp. 408–411. doi:10.1109/ICMLA.2017.0-127.

[20] J. Risch, R. Krestel, Domain-specific Word Embeddings for Patent Classification, Data Technologies and Applications 53 (2019) 108–122. doi:10.1108/DTA-01-2019-0002.

[21] S. Althammer, M. Buckley, S. Hofstätter, A. Hanbury, Linguistically informed masking for representation learning in the patent domain, in: Proceedings of the 2nd Workshop on Patent Text Mining and Semantic Technologies (PatentSemTech) 2021 co-located with the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'21), Online, 2021.

[22] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, J. Eisenstein, Explainable Prediction of Medical Codes from Clinical Text, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'18), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 1101–1111. doi:10.18653/v1/N18-1100.

[23] A. Rios, R. Kavuluru, Few-Shot and Zero-Shot Multi-Label Learning for Structured Label Spaces, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP'18), Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3132–3142. doi:10.18653/v1/D18-1352.

[24] T. N. Kipf, M. Welling, Semi-Supervised Classification with Graph Convolutional Networks, in: Proceedings of the 5th International Conference on Learning Representations (ICLR'17), Toulon, France, 2017. URL: https://openreview.net/forum?id=SJU4ayYgl.

[25] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, I. Androutsopoulos, Large-Scale Multi-Label

Text Classification on EU Legislation, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL'19), Association for Computational Linguistics, Florence, Italy, 2019, pp. 6314–6322. doi:10.18653/v1/P19-1636.

[26] A. Johnson, T. Pollard, L. Shen, L.-w. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Celi, R. Mark, MIMIC-III, a Freely Accessible Critical Care Database, Scientific Data 3 (2016) 160035. doi:10.1038/sdata.2016.35.

[27] D. D. Lewis, Y. Yang, T. G. Rose, F. Li, RCV1: A New Benchmark Collection for Text Categorization Research, J. Mach. Learn. Res. 5 (2004) 361–397. URL: http://jmlr.org/papers/volume5/lewis04a/lewis04a.pdf.

[28] S. Kiritchenko, S. Matwin, A. F. Famili, Functional Annotation of Genes Using Hierarchical Text Categorization, in: Proceedings of the BioLINK SIG: Linking Literature, Information and Knowledge for Biology, 2005. URL: https://www.site.uottawa.ca/~stan/papers/2004/p15.pdf.

[29] C. Cortes, V. Vapnik, Support-Vector Networks, Machine Learning 20 (1995) 273–297. doi:10.1007/BF00994018.

[30] F. Chollet, et al., Keras, https://github.com/fchollet/keras, 2015.

[31] 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, A. M. Rush, HuggingFace's Transformers: State-of-the-art Natural Language Processing, 2019. arXiv:1910.03771.

[32] L. Lange, H. Adel, J. Strötgen, D. Klakow, FAME: Feature-Based Adversarial Meta-Embeddings for Robust Input Representations, in: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP'21), Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2021, pp. 8382–8395. doi:10.18653/v1/2021.emnlp-main.660.

[33] I. Beltagy, M. E. Peters, A. Cohan, Longformer: The Long-Document Transformer, 2020. arXiv:2004.05150.