Intellectual Property Office

The use of PatentBERT for the allocation of patent applications in the UK IPO

Colm McKenna, Howard Chen, Mark Edwards

Objective & Approach

Objective:

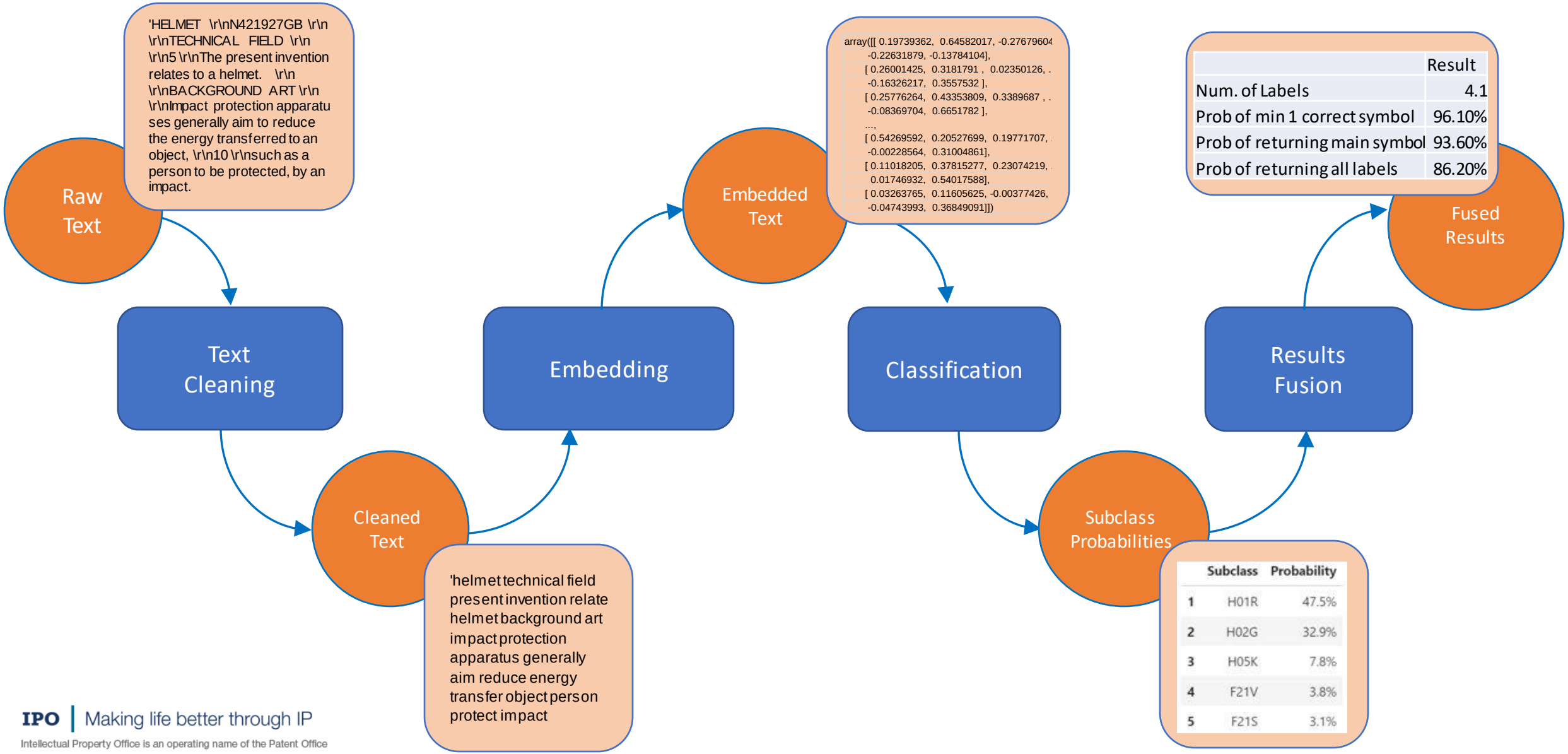
Develop an AI powered classification tool to accurately allocate patent applications (at the subclass level) to the appropriate patent examiner groups.

Techniques used:

- Text Cleaning
- Text Embedding
- Classification
- Classification Results Fusion

Intellectual Property Office

Patent Allocation Data Flow

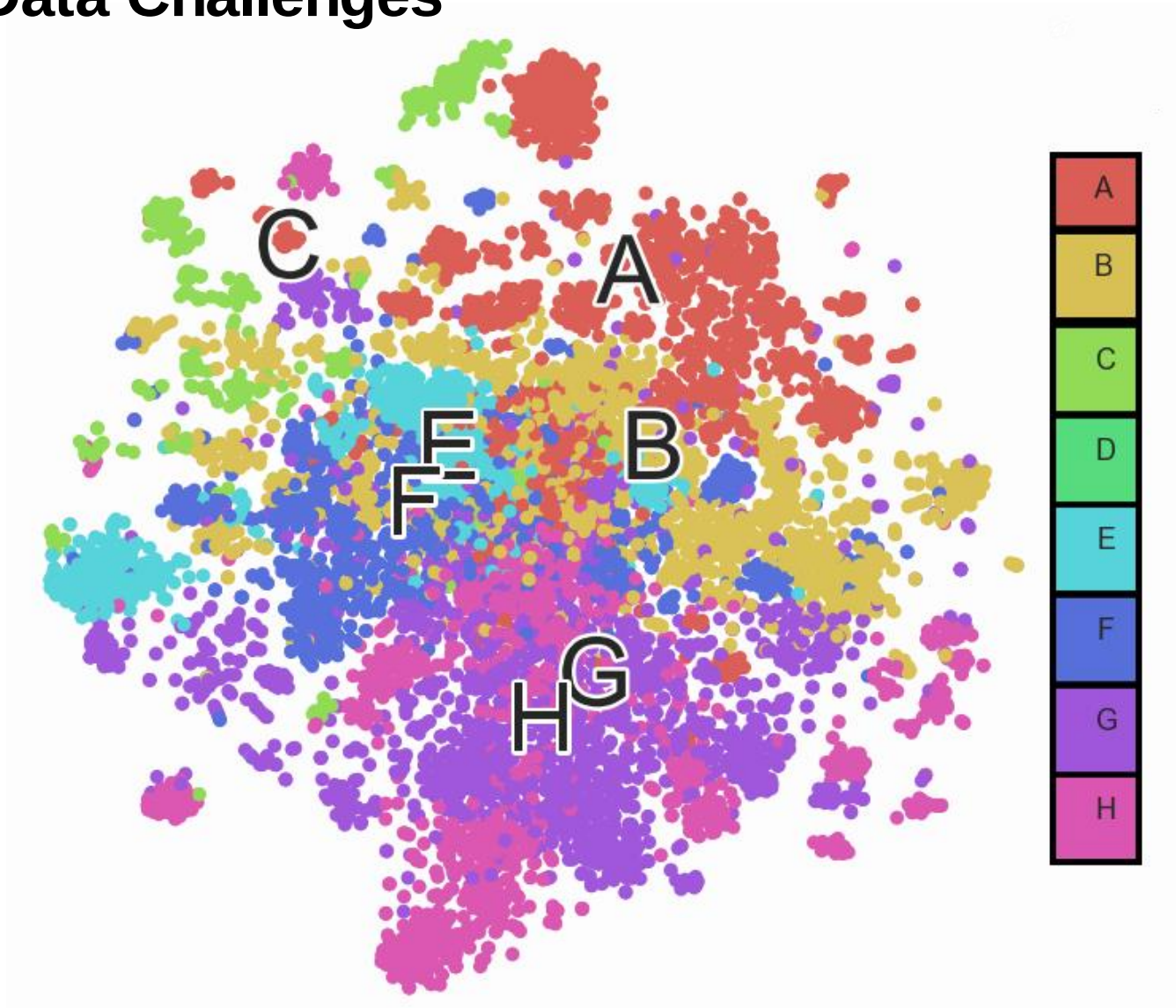


Explanation of terms

- The term we will use is classification **symbols** when discussing applications, but **labels** in reference to machine learning.
- Thus, anything predicted by our model will be a **label**, while the ground truth for an application will be the classification **symbols** applied by an examiner.
- This separation will become important when we discuss the different types of output from our models (such as multiclass and multilabel).

Data Challenges

- Illustration of the difficulty of automating classification - 10,000 patent applications shown – sections displayed in colours in key.
- Displayed using t-distributed stochastic neighbour embedding (t-SNE) – reduces data with high dimensionality to 2D
- While some sections are separated from others, many have **substantial overlap**.



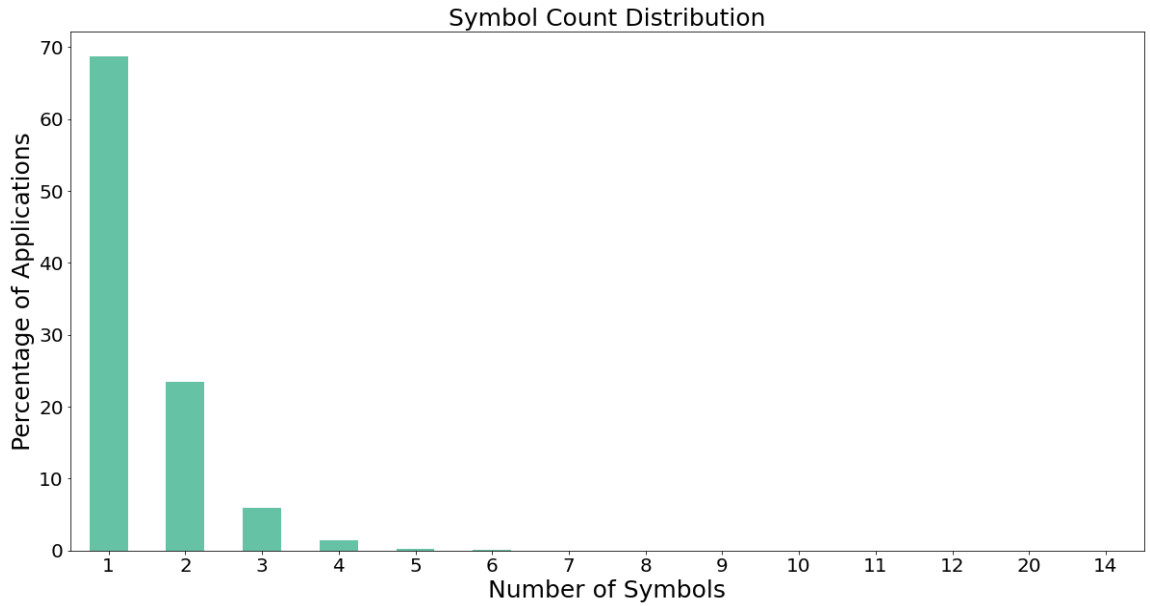
Dataset Characteristics

The dataset that used is composed of 584,826 UK patent application description documents. Each document is labelled with at least one IPC symbol denoting a subclass.

- Total number of documents: 584,826, Number of Subclasses covered as primary symbol: 656

Distribution of number of symbols

- Distribution of the number of unique symbols applied to applications in our dataset.
- Majority of applications have a primary symbol only
- We will discuss the significance of patents application having one symbol or more than one symbol later in this presentation - a different classification architecture is required.



Text Cleaning

- The text cleaning process varies depending on the embedding to be used:
  - PatentBERT Embedding
    - Very light cleaning only
      - Remove punctuation
      - Make lower case
    - Context must be retained
  - Traditional Embedding -TFIDF etc
    - Heavier cleaning required
      - Normalising – lowercase, remove punctuation
      - Stemming - Reduce words to their stem (e.g., connection, connected, and connecting -> connect)
      - Lemmatising - Reduced words to their base form, considers context (meeting -> met, was -> be, mice -> mouse)

Confirming Classification Symbols are Current and Valid

- Using data with symbols which are no longer correct reduces the accuracy the ML model can achieve
- Therefore, we tracked how symbols have changed (updating if required) using:
  - **Concordance file** available from **WIPO** which specify how symbols change between editions and
  - **Validity files** to identify whether a symbol is still used

- This allows us to build a change tree for each application,
- **Changes** to the primary symbol are **tracked** from year of filing.
- Can become very large and unwieldy as shown

Change Tree

```
19850101: A61M0001000000
  |--19850101: A61B0005140000
    |--20000101: A61B0005140000
      |--20000101: A61B0005145000
      |  |--20060101: A61B0005145000
      |    |--20060101: A61B0005145500
      |    |--20060101: A61B0005145900
      |    |--20060101: A61B0005146400
      |    |--20060101: A61B0005146800
      |    |--20060101: A61B0005147300
      |    |--20060101: A61B0005147700
      |    |--20060101: A61B0005148200
      |    |--20060101: A61B0005148600
      |    |--20060101: A61B0005149100
      |    |--20060101: A61B0005149500
      |--20000101: A61B0005150000
      |  |--20060101: A61B0005150000
      |    |--20060101: A61B0005151000
      |    |--20060101: A61B0005153000
      |    |--20060101: A61B0005154000
      |    |--20060101: A61B0005157000
      |--20000101: A61B0005155000
```

- A 'pruning' algorithm was developed
- Starts at current symbol
- **Merges branches** if they **end with same symbol**
- Algorithm then moves up a level and repeats
- Pruned tree is clearer and shows symbol is now **A61B**
- We **exclude applications** where a symbol has been **mapped to symbols in multiple subclasses.**

Pruned Change Tree

```
19850101:     A61M0001000000
  |--19850101: A61B
```

Embedding

- Machine learning algorithms work with vectors rather than text.
- Therefore, our **text** must be **converted** to **vectors** – a **point in multidimensional space.**
- Embedding is the process of converting text to vectors.

Bag of Words:

- Bag of Words – count of words in document
- Bag of N-Grams – chunks of n contiguous words
- Term frequency, inverse document frequency (TF-IDF)
  - Reduces weight of words common to more documents

Bag of Words Representation

	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0

Includes the Context of Words:

- Word2Vec
  - **Related words** placed **close** together in **vector space**
- Doc2Vec
  - Learns word vectors **and** document vector

BERT (Bidirectional Encoder Representations from Transformers)

- Open-source ML framework for natural language processing (NLP)
- Helps computers understand ambiguous language by using surrounding text to establish meaning
- Generates **different vector** for **same word** where **meaning is different**, e.g.,
  - "The man was accused of robbing a **bank**."
  - "The man went fishing by the **bank** of the river."

General Bert:

- Published in 2018 by Google, open sourced, used in google search engine
- Pre-trained (no labels) on 800M words from BooksCorpus and 2,500M words from English Wikipedia
- Maximum input sequence length: 512 tokens

PatentBert:

- Published in 2020 by Google, open sourced
- **Fine-tunes** a pre-trained **BERT model** for **patent classification**
- Trained on >100 million patent documents including abstract, claims, description

Comparison of Embedding Methods

Embedding	Embedding Remark	Classifier Type	Data Size	Number of Subclasses	Prediction Accuracy
Word2vec	vector_size=300, sg=0	MLP	135286	336	61.2
Doc2Vec	vector_size=300, Distributed Bag of Words				53.3
TF-IDF	10000 features				65.59
w2v Full Text	vector_size=300, sg=0	MLP	8807	212	49.4
w2v Full Text	vector_size=300, bigrams, sg=0				47.5
Bert Patent	512 Tokens	MLP	153983	616	67.7
TF-IDF	10000 features				67.2

Classification Architectures Tested

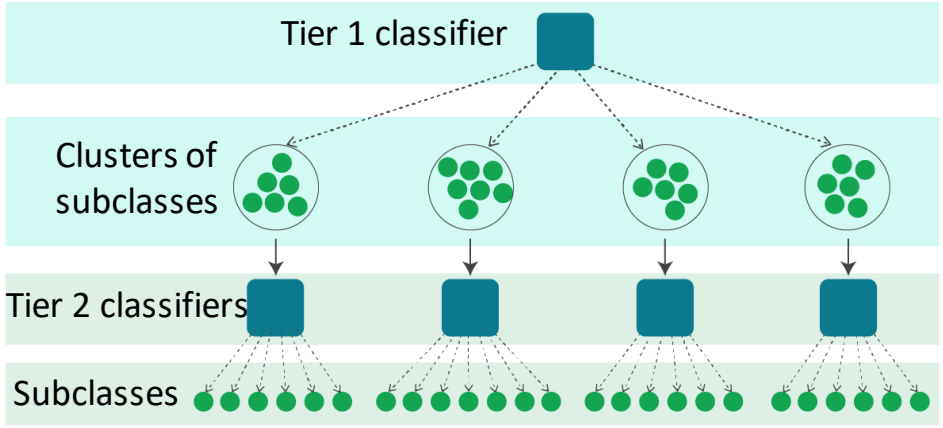
- **2 Tier Classification**
  - **Tier 1** consists of a **single classifier** trained with documents **labelled by cluster** – not subclass.
  - **Tier 2** consists of **one classifier** for **each cluster –** only trained with docs from that cluster
  - Tier 2 classifier will assign each document to a subclass.

- **Binary Classification**
  - Classifier is trained with all documents.
  - A **single subclass** is **labelled 1**, **all others labelled 0**.
  - **Highly imbalanced dataset**

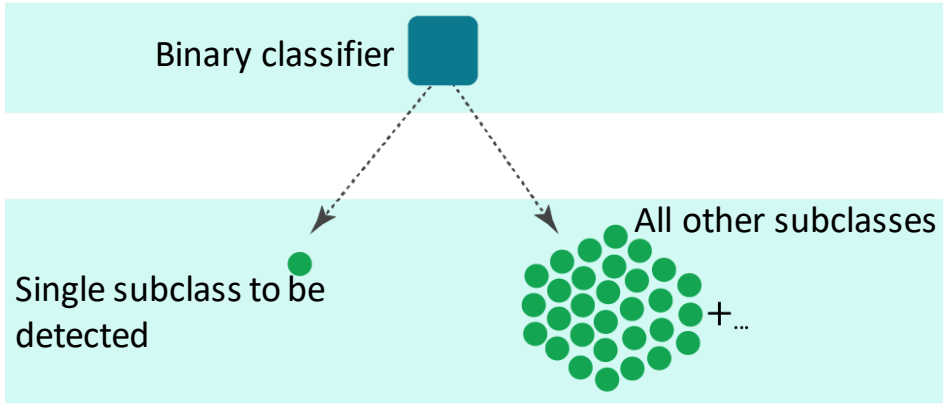
- **Results**
  - **2 Tier** – two-step process resulted in **lower accuracy**
  - Binary - may assist identifying misclassification in very specific cases

- **Universal Classifier**
  - **Universal classifier** - single classifier assigning to subclass level in one step
  - Produces higher accuracy than other architectures.
  - **Therefore, universal classifier architecture is used in following work**

2 Tier Classification



Binary Classification



Comparison of Classification Models

Embedding	Embedding Remark	Classifier Type	Data Size	Number of Subclasses	Prediction Accuracy
TF-IDF	min_df=0.0, max_df=1.0, ngram_range=(1,1)	XGBoosts	8809	212	43
		Random Forest			37.9
Word2vec	vector_size=300, sg=0	SGD	135286	336	36.7
		XGBoosts			39.4
		Neural Network			61.2
Doc2Vec	vector_size=300,Distributed Bag of Words	SGD	135286	336	47.1
		Logistic Regression			49.7
		Neural Network			53.3

Multiclass vs Multilabel Models

- Multiclass model typically used to classify data which fits in one class only
- Multilabel model used to classify data which may require more than one label
- Multi-class probabilities sum to 1
- Multi-label probabilities are independent

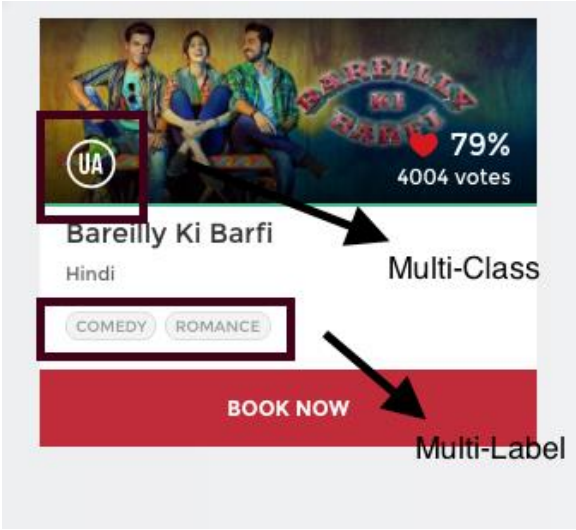


Illustration of Multiclass vs multilabel in film classification and film genres respectively.

Multi-class and Multi-label Classifications Results

Typical Multi-class Results

	Subclass	Probability
Main Label	C07D	99.25%

+

Typical Multi-label Results

Number of output labels varies according to the Confidence Threshold selected

	Subclass	Probability
Label 1	C07D	99.63% %
Label 2	A61P	96.78% %
Label 3	A61K	96.78% %

=

Fused Results

Predictions:

	Subclass	Probability
Main Label	C07D	99.63
Additional Label 1	A61P	96.78
Additional Label 2	A61K	96.78

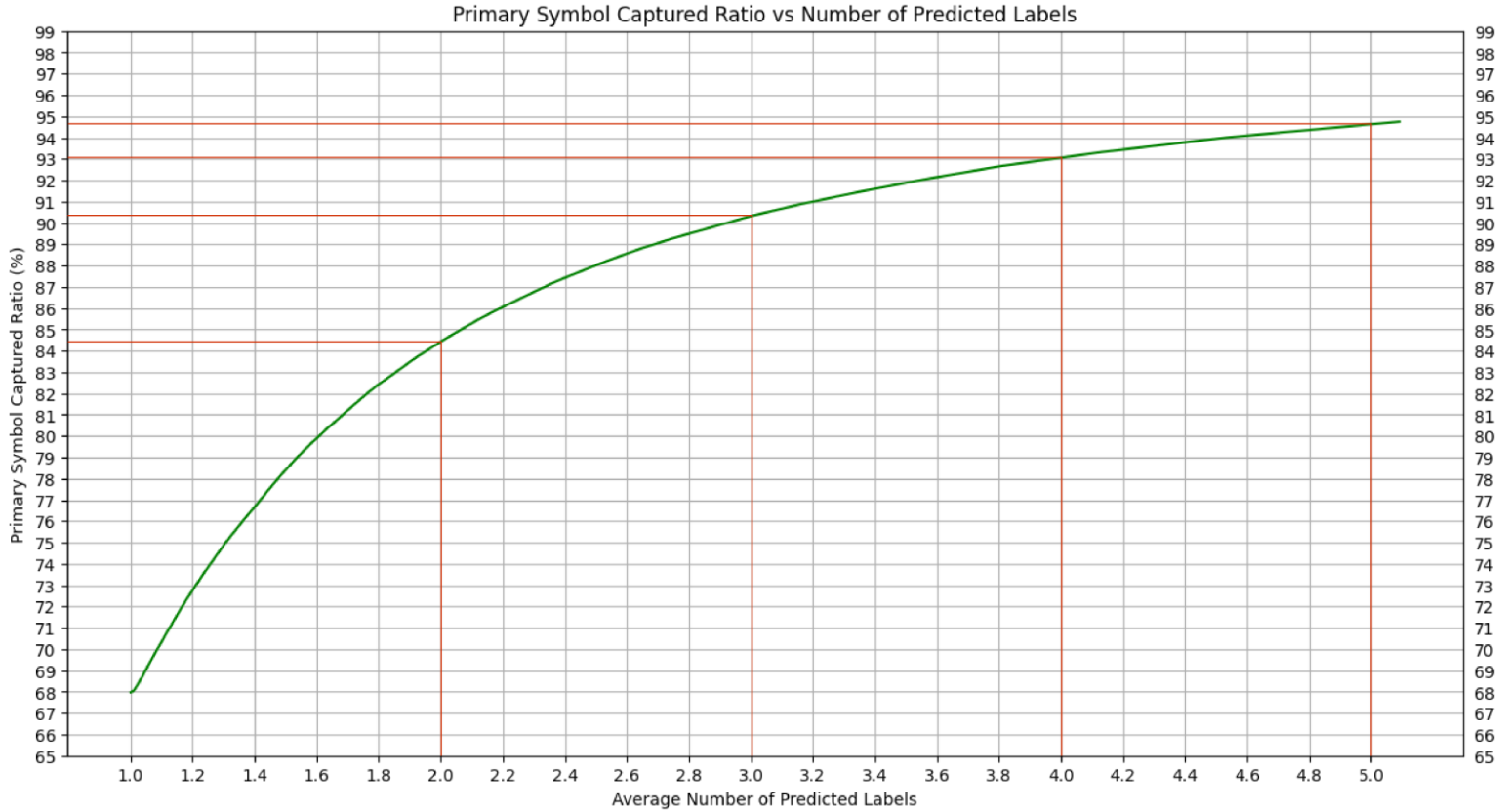
Multi-Label Classification Results Evaluation

- Multiclass and Multilabel results were fused
- A confidence threshold was set at 6.5%
- The table shows the results achieved
- The **average number of symbols returned is 4.1** and the **probability** that the **primary symbol is included = 93.6%**

	Result
Average Num. of Labels	4.1
Prob. returning minimum of 1 correct Symbol	96.1%
Prob. of returning Primary Symbol	93.6%
Prob. of all Symbols are captured	86.2%

Fused Results – Capture Ratio vs Number of Labels

- This graph shows the how the probability of returning the primary symbol increases with the number of symbols returned



Establish baseline accuracy by document type

- Approx. 100k documents have claims, description and abstract.
- Train a model using each data type
- Generate a ‘baseline’ accuracy that we can use to determine the performance of different models and combinations
- These baseline values are shown in the table.
- We will now discuss how we can combine multiclass and multilabel to produce a better model

Baseline Accuracy by Data type

Document Type	Dataset Size	Primary Symbol Accuracy
Claims	~100K	60.67%
Description	~540K	68.54%
Abstract	~100K	69.56%

Primary Symbol Prediction Accuracy

- The table shows how fusing multiclass and multilabel results improves the prediction accuracy.
- Abstracts produce the highest accuracy of any single document type.
- Fused description and abstract documents produce the highest accuracy for fused documents

Table showing the accuracy of primary symbol prediction for single/fused results and a range of document types

	Type of Combination	
Data type	**Multiclass (i.e., base accuracy)**	**Multiclass & Multilabel**
Claims	60.67%	62.74%
Description	65.81%	66.99%
Abstract	69.56%	71.21%
Description & Abstract	72.79%	73.14%
Claims & Description	68.67%	69.19%
Claims & Abstract	71.02%	71.34%
Claims, Description and Abstract	72.44%	72.79%

Questions & Answers