# Domain Adaptation via Adaptive Tokenization

**Eric Khiu**
University of Michigan, Ann Arbor
erickhiu@umich.edu

**Cassandra Goh**
University of Michigan, Ann Arbor
kexuan@umich.edu

## 1 Problem Description

The advancement of Pretrained Language Models (PLMs) has revolutionized Natural Language Processing (NLP), with models like RoBERTa (Liu et al., 2019) being trained on extensive and diverse corpora, including text from encyclopedias, news, literature, and web content (Raffel et al., 2020; Yang et al., 2019). These models typically undergo two crucial training stages: *pretraining* through unsupervised learning on a base corpus, and *fine-tuning* on specific supervised tasks using labeled data.

Despite their robustness, PLMs face challenges in domain adaptation, particularly when applied to contexts distinct from their training data, leading to performance drops (Elsahar and Gallé, 2019), possibly due to the lack of vocabulary or syntax (Figure 1). Strategies like *domain-adaptive pretraining* (DAPT) and *task-adaptive pretraining* (TAPT), though useful, often fall short of models trained specifically on in-domain data (Gururangan et al., 2020). An efficient alternative, adaptive tokenization, has shown promising results in maintaining the benefits of domain-specific pretraining (Sachidananda et al., 2021).

As described in Gururangan et al., 2020, *domain* refers to "a distribution over language characterizing a given topic or genre". We wonder if the domain adaptation task, which has been shown to be effective to improve the performance of PLMs (Sachidananda et al., 2021; Gururangan et al., 2020), is analogous to genre adaptation in NLP task related to popular forms of entertainment, such as movies, TV shows, books, etc. Hence, our project focuses on a smaller scale of the domain adaptation task, i.e., movie genre adaptation, due to the limitation in computing power to handle huge amount of data across different domains. The NLP task chosen for this project is a binary classification task on the helpfulness of the movie review, as described in Section 3.1. We employed the RoBERTa classifier with a genre-adapted tokenizer to perform this task, and evaluated the model's performance based on accuracy and F1-score.

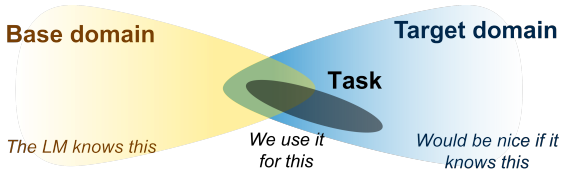


Figure 1: Illustration of data distribution. *Base domain*: domain of corpora that the LM is trained on; *target domain*: domain that the LM is intended to used to perform the *task*.

In this project, we have successfully implemented and assessed the impact of an Adaptive Tokenizer (AT) in the context of genre-specific language processing. Our results, detailed Section 4.2, show significant improvements in accuracy and F1-score with AT, especially without tf-idf, compared to the RoBERTa-base tokenizer.

## 2 Related Work

Gururangan et al., 2020 introduces domain-adaptive pretraining (DAPT) and task-adaptive pretraining (TAPT), which involve using either a distinct unlabeled domain-specific corpus or a task's training data for further pretraining a language model. These methods involve further pretraining of a language model using either domain-specific corpora or a task's training data, showing clear benefits over mere fine-tuning of PLMs. This concept inspired Sachidananda et al., 2021 to adopt in-domain corpora for better model adaptation.

Another key development by Hofmann et al., 2021 pointed out the limitations of Wordpiece tokenization in capturing the semantic nuances of complex words. They found that subword tokenization, accounting for linguistic prefixes and suffixes,

was more effective in understanding word polarity and domain. This approach, involving novel embedding techniques within BERT, aligns with the token sequence selection methods used by Sachidananda et al., 2021.

In the calculation of domain shift score, extensive studies on identifying domain-characteristic words have been done, with numerous metrics introduced for this purpose, involving the comparison of token distributions across different corpora, as demonstrated by the work of Rayson et al., 1997, Monroe et al., 2008, Kessler, 2017, and Muthukrishnan et al., 2008 employed the pointwise KL-divergence as a method to differentiate the informativeness of candidate key phrases.

Building on the adaptive tokenization approach by Sachidananda et al., 2021, we included tf-idf in the calculation for domain shift score, grounded in distributional hypothesis to further refine the domain (genre) adaptation process. The rationale behind this integration is to enhance the significance of tokens that are particularly effective in distinguishing between different genres during the adaptation process. Our contribution lies in the novel integration of vector semantics with adaptive tokenization, providing new insights into domain adaptation in NLP. This approach paves the way for future model development, offering a nuanced understanding of genre-specific language processing in PLMs.

## 3 Methodology

### 3.1 Dataset

In this project, we used the IMDb movie review dataset, sourced from Pal et al., 2020, which includes around 1 million unique reviews. Initially covering 17 movie genres, we specifically focused on four genres (number of reviews in parenthesis): action (2495), comedy (2493), crime (2496), and romance (2499). This selection allows us to examine the efficacy of our genre-adaptation method across diverse cinematic styles and narratives. The diversity of genres allows us to assess the effectiveness of our genre-adaptation method. An example of movie review in this dataset is shown in Figure 2.

**Data Preprocessing** The extraction of movie reviews from the dataset was conducted using Beautifulsoup (Richardson, 2007), a robust Python library for web scraping. During this process, we paid special attention to data cleaning, particularly

redbullrex 22 January 2021

If someone else acted as Joker, I would give the movie 7-8 stars. The majority of people ended up loving the villain more than the hero, and that rarely happends in movies. Rest in peace Heath Ledger.

123 out of 151 found this helpful. Was this review helpful? Sign in to vote.

Figure 2: Example of movie review in our dataset

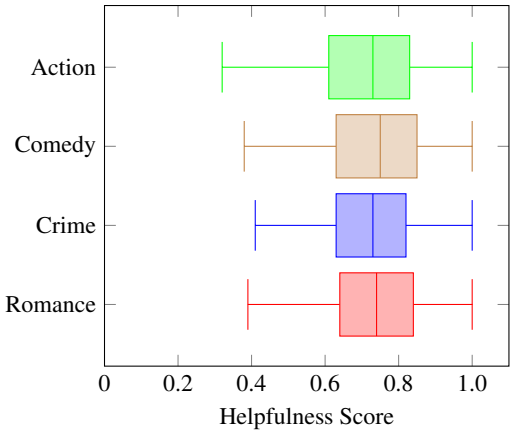


Figure 3: Distribution of helpfulness score of the movie reviews in each genre in our dataset.

the removal of backslashes that were remnants of HTML formatting. These elements were deemed irrelevant to the content of the movie reviews and therefore excluded to maintain the integrity of the data.

**Dataset Properties** A critical aspect of this dataset is the *helpfulness score* in the form of "$x$ out of $y$ found this helpful". We define the *helpfulness score* as the ratio $x/y$. This feature enables us to undertake the binary classification task in our project. However, we noticed that our dataset is biased toward "helpful", as shown in Figure 3. The mean helpfulness scores for action, comedy, crime, and romance are 0.7277, 0.7425, 0.7315, 0.7428. Recognizing the biases, we set the threshold for a helpful movie review as 0.8.

**Data Organization** We organized our data as a dictionary, where the keys are the genre, and the values are in the form {`review_content, helpfulness, label, idx`} where `label` is 1 for helpful review and 0 for unhelpful review. We stored the data in four .json files, which were then used for adaptive tokenization, fine-tuning, validation, and testing.

### 3.2 Adaptive Tokenization

Following Sachidananda et al., 2021, our adaptive tokenization algorithm, as shown in Figure 4, consists of the following four steps:

1. Compute token sequence distribution
2. Calculate domain shift of token sequence
3. Select token sequences to be augmented
4. Augment token sequences embeddings in base tokenizer

This algorithm aims to expand the base tokenizer by adding new in-domain tokens, which are sequences that build upon existing token sequences present in the in-domain corpus. The embeddings for those token sequences are generated in step 4. We proposed to (1) include more hyperparameters in the first part of the algorithm as well as providing an alternative to calculate the probability distribution and (2) modify the second part of this algorithm to include the Term Frequency - Inverse Document Frequency (tf-idf) scores for each token sequence when computing the domain shift score.

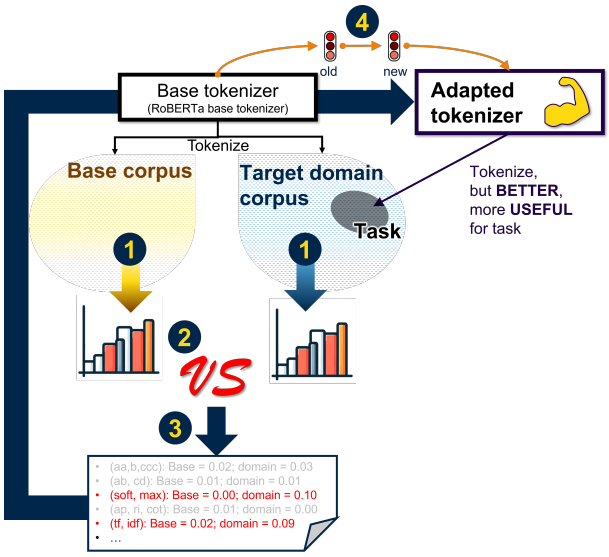


Figure 4: Our adaptive tokenization architecture for domain (genre) adaptation.

#### 3.2.1 Token Sequence Distribution

Algorithm 1 shows our modified pseudocode from Sachidananda et al., 2021 to compute the token sequence distribution. We first tokenize the words in the corpus using RoBERTa-base tokenizer. For each token sequence $s$ in corpus $C$, we record its count as $C_s$. We only consider token sequences that satisfy the following constraints:

- The sequence is limited to full words determined by white spaces
- The sequence length is at most 10
- The sequence frequency is at least 20 in the corpus.

Next, we normalize the sequence distribution by dividing by the total number of words in the corpus. Finally, we estimate how much a token sequence, $C_s$, resembles a phrase by calculating a probability, $P_C(s)$. It is calculated using the following formula in Sachidananda et al., 2021

$$P_C(s) = \frac{C_s}{C_t}$$

where $t$ is the first $|s| - 1$ sequence. The resulting probabilities indicate how unexpected or "phrase-like" the sequence $s$ appears in the corpus. We refer this probability distribution as *empirical distribution*. In our implementation, we proposed a parameter $l$ such that $t = |s| - l$.

To calculate the probability distribution, we also provide an alternative to calculate using the SOFTMAX function. This is activated by passing in the parameter $SM$ to be `true`.

Using Algorithm 1, we calculated the probability distribution for the base corpus BOOKCORPUS (Zhu et al., 2015), that is used in the pretraining of ROBERTA, and the probability distribution for the domain-specific corpora. The domain shift scores between two token sequences distributions ($P_{base}$ and $P_{domain}$) are calculated in the next section.

#### 3.2.2 Domain Shift Scoring with Combined KL Divergence and TF-IDF Scores

To calculate the domain shift score for a sequence $s$, we incorporate TF-IDF scores on top of pointwise KL-Divergence considered in Sachidananda et al., 2021. In our implementation, the domain shift score is given by

$$R(s) = \lambda_1 D_{KL}(P_D(s)||P_S(s)) + \lambda_2 T_D(s)$$

where

$$D_{KL}(P_D(s)||P_S(s)) = P_D(s) \log \frac{P_D(s)}{P_S(s)}$$

KL divergence reflects how the likelihood of a sequence $s$ being phrase-like in the target domain corpus $D$ deviates from its baseline likelihood in the source corpus $C$. Note that this divergence

**Require:** Base tokenizer $Tok$, Base and Domain Unigram Distributions $U_{base}, U_{domain}$, max sequence length $m$, minimum sequence frequency $f$, number of tokens to drop $l$, boolean to use softmax $SM$

**function** TOKENSEQDIST($Tok, U, m, f, l$)
  **for** word, count $(w, count)$ in $U$ **do** ▷ Subword sequence limited to full words
    $Seq[t_0 : t_n] \leftarrow Tok(w)$
    **if** $n > m$ **then** ▷ If the word contains more than $m$ subword tokens
      **continue**
    **end if**
    **for** $i = 1, ..., n$ **do**
      $T[Seq[: i]] += count$
    **end for**
  **end for**
  Remove sequences with frequency $< f$ from $T$
  $T$.values() /= sum ($U$.values()) ▷ Normalize Sequence Distribution
  Initialize $P$ for token sequence distribution
  **if** $SM$ **then**
    $P \leftarrow$ SOFTMAX($T$)
  **else**
    **for** $Seq$ in $T$ **do**
      $DropLastL \leftarrow$ first $|Seq| - l$ tokens in $Seq$
      $P[Seq] \leftarrow \frac{T[Seq]}{T[DropLastL]}$
    **end for**
  **end if**
  **return** $P$
**end function**
$P_{base} \leftarrow$ TOKENSEQDIST($Tok, U_{base}$)
$P_{domain} \leftarrow$ TOKENSEQDIST($Tok, U_{domain}$)

Algorithm 1: Computing token sequence distribution.

**Require:** Base and domain token sequences distributions $P_{base}, P_{domain}$, target domain corpus $D$, divergence weights $\lambda_1, \lambda_2$

**function** DOMAINSHIFT($P_{base}, P_{domain}, \lambda_1, \lambda_2$)
  Initialize $DS\_Scores$ for domain shift scores
  **for** $Seq$ in $P_{base} \cap P_{domain}$ **do**
    $D_{KL} \leftarrow P_{domain}[Seq] \cdot \log \frac{P_{domain}[Seq]}{P_{base}[Seq]}$
    $TF\_IDF \leftarrow$ TF-IDF($Seq, D$)
    $DS\_Scores[Seq] \leftarrow \lambda_1 D_{KL} + \lambda_2 TF\_IDF$
  **end for**
  **return** $DS\_Scores$
**end function**

Algorithm 2: Computing domain shift scores.

measure is not symmetric: higher distribution in the target domain $D$ will result in a higher divergence score, informing the importance of a certain sequence to be included in the domain adaptation. TF-IDF score, on the other hand, is to measure how important the sequence is to the target domain corpus. This combined domain shift score gives a holistic measure of domain shift by considering both the distributional differences (pointwise KL divergence) and term importance (TF-IDF).

#### 3.2.3 Token Sequences Selection

Similar to Sachidananda et al., 2021, we incorporate the top $N$ sequences with the highest domain shift scores into the domain-enhanced tokenizer. See Algorithm 3 for implementation.

#### 3.2.4 Embeddings Augmentation

We adapt **subword-based initialization.** (Casanueva et al., 2020) to augment the subword embeddings in the tokenizer. This is done by adding the selected token sequences as new token, embedded as the mean of their RoBERTa-based fixed subword embeddings, as described in Algorithm 4.

## 4 Experiments

After adaptive tokenization described in Section 3.2, we proceeded with the binary classification task. This is done by initializing the tokenizer with augmented token embeddings generated in Section 3.2.4 into `RobertaForSequenceClassification` (Liu et al., 2019). Note that we freeze the weight parameters in this model, and only train the head of the model to perform binary classification. Since RoBERTa-base has 12 pre-trained transformer layers, it still have 12 transformer layers after freezing the weight parameters. We first trained the model on the crime genre, using the validation set, we fine-tuned the hyperparameter to obtain the best learning rate and number of epoch. To save implementation time, we use this best learning rate and number of epoch to train and evaluate the model on prediciting the label for the other three genres. Finally, we evaluated the models' performance based on their accuracy and F1-score, as in Section 4.2

### 4.1 Experimental Setup

We allocated our data as follows: 40% of the reviews were used for tokenizer adaptation, another 40% for fine tuning for classification task, 10% for validation purposes, and the remaining 10% for testing the model. This distribution was meticulously planned to optimize the training process while ensuring robust model evaluation and validation.

Using the 10% data allocated for validation, we performed hyperparameter search using grid search. The list of search values for each hyperparameter are as follows:

- Learning rate: [1e-4, 5e-5, 1e-5]
- Batch size: [16]
- Number of epochs: [3,5,8]
- Distribution: [`empirical`, `softmax`]
  - For `empirical`, number of tokens to drop (line 18, Algorithm 1), $l$: [1, 2, 3]
- Number of augmentation, $N$: [5000, 10000, 20000]

Note that due to time and computing power constraints, we only used batch size = 16. We also did not manage to perform stochastic gradient descent for hyperparameter seach for $\lambda_1, \lambda_2$ (weights of KL-divergence and tf-idf in domain shift score calculation). Consequently, we employed AT without tf-idf ($\lambda_1 = 1, \lambda_2 = 0$) and AT with tf-idf ($\lambda_1 = 0.5, \lambda_2 = 0.5$).

Table 1 records the best hyperparameter settings for genre using AT without and with tf-idf in our experiments. Notably, for AT without tf-idf, three out of the four genres exhibit improved performance when the `softmax` function is used to calculate the distribution. For the genre that fares better with the empirical distribution, the introduced hyperparameter—the number of tokens to be dropped—also influences the results. Optimal hyperparameter settings for 1 are either 2 or 3, in contrast to 1 as suggested in Sachidananda et al., 2021.

For the binary classification task, we used cross entropy loss as our lost function. The best learning date was 0.0001 whereas the best number of epoch was 3.

### 4.2 Results

In this section, we examine the effectiveness of our Adaptive Tokenizer (AT) in enhancing genre-specific vocabulary, as detailed in Table 2. We then shift our focus to the performance of the AT

**Require:** Domain shift scores $DS\_Scores$, number of augmentation to make $N$

**function** SEQUENCESELECTION($DS\_Scores$)
  Initialize $Aug$ for sequences to be augmented
  SORTDESCENDING($DS\_Scores$)
  **for** $Seq$ in $DS\_Scores$ **do**
    **if** LEN($Aug$) $= N$ **then**
      **break**
    **end if**
    $Aug \leftarrow Aug \cup \{Seq\}$
  **end for**
  **return** $Aug$
**end function**

Algorithm 3: Selecting sequences to be augmented.

**Require:** Base tokenizer $Tok$, token sequences to be augmented $Aug$

**function** SUBWORD-BASEDINIT($Tok, Aug$)
  **for** $Seq$ in $Aug$ **do**
    $Embd \leftarrow$ MEAN($Tok[tokens$ in $Seq]$)
  **end for**
  $Tok \leftarrow Tok \cup \{(Seq, Embd)\}$
**end function**

Algorithm 4: Subword-based initialization

| Tokenizer | Genre | Distribution | # augmentation, $N$ |
|---|---|---|---|
| AT (without tf-idf) | Action | `empirical` (3) | 20000 |
| | Comedy | `softmax` | 5000 |
| | Crime | `softmax` | 20000 |
| | Romance | `softmax` | 5000 |
| AT (with tf-idf) | Action | `empirical` (3) | 20000 |
| | Comedy | `empirical` (2) | 10000 |
| | Crime | `empirical` (2) | 10000 |
| | Romance | `softmax` | 5000 |

Table 1: Best hyperparameter settings for each genre for AT (without tf-idf) and AT (with tf-idf). If empirical distribution is used, number in parentheses indicates number of tokens to drop, $l$.

in helpfulness classification, comparing it with the ROBERTA-base tokenizer baseline across four genres. The analysis, illustrated in Figures 5 and 6, evaluates the improvements in both accuracy and F1-score, highlighting AT's particularly strong impact in AT without tf-idf.

#### 4.2.1 Adaptive Tokenization

In Table 2, we present examples of augmented vocabulary selected by our AT for each of the four genre in our experiments. Notice that in each genre, the token sequences identified by the AT is related to the theme of the corresponding genre. For example, in the comedy genre, tokens like "animals" from "anim" and "als", and "superhero" from "super" and "hero" are representative of the humorous and light-hearted narratives typical of this genre, while in the Crime genre, token sequences that reflect its gritty and suspenseful nature with words like "casino" from "cas" and "ino", and "sus" from "s", "us" were selected.

#### 4.2.2 Helpfulness Classification

**Baseline** In our experiments, we used the ROBERTA-base tokenizer as our baseline. This means the classifier perform the task without any domain adaptation via adaptive tokenization. For the baseline model, the average accuracy across four genres in our experiment was 0.4434, whereas the average F1-score was 0.5080.

**Accuracy** Figure 5 illustrates the accuracy achieved by each tokenizer across the different genres. Notably, the Adaptive Tokenizer (without tf-idf) outperformed the ROBERTA-base tokenizer in all genres, with the most significant improvement observed in the comedy genre, where accuracy increased from 0.40 to 0.61. The use of AT with tf-idf also showed improvement over the base tokenizer but was marginally less effective than AT without tf-idf, except in the crime genre, where the base tokenizer outperformed AT with tf-idf.

| Action | Comedy | Crime | Romance |
|---|---|---|---|
| ('app', 'reci') → appreci | ('anim', 'als') → animals | ('cas', 'ino') → casino | ('touch', 'ing') → touching |
| ('pres', 'ence') → presence | ('super', 'hero') → superhero | ('over', '-', 'the', '-', 'top') → over-the-top | ('pres', 'ence') → presence |
| ('out', 'standing') → outstanding | ('ch', 'arming') → charming | ('s', 'us') → sus | ('chem', 'istry') → chemistry |
| ('fin', 'ale') → finale | ('sat', 'ire') → satire | ('c', 'ops') → cops | ('m', 'oves') → moves |
| ('sp', 'y') → spy | ('laugh', 'ing') → laughing | ('gang', 'sters') → gangsters | ('rom', 'antic') → romantic |

Table 2: Examples of token sequence with high domain shift scores between base and domain corpora sequence distribution.

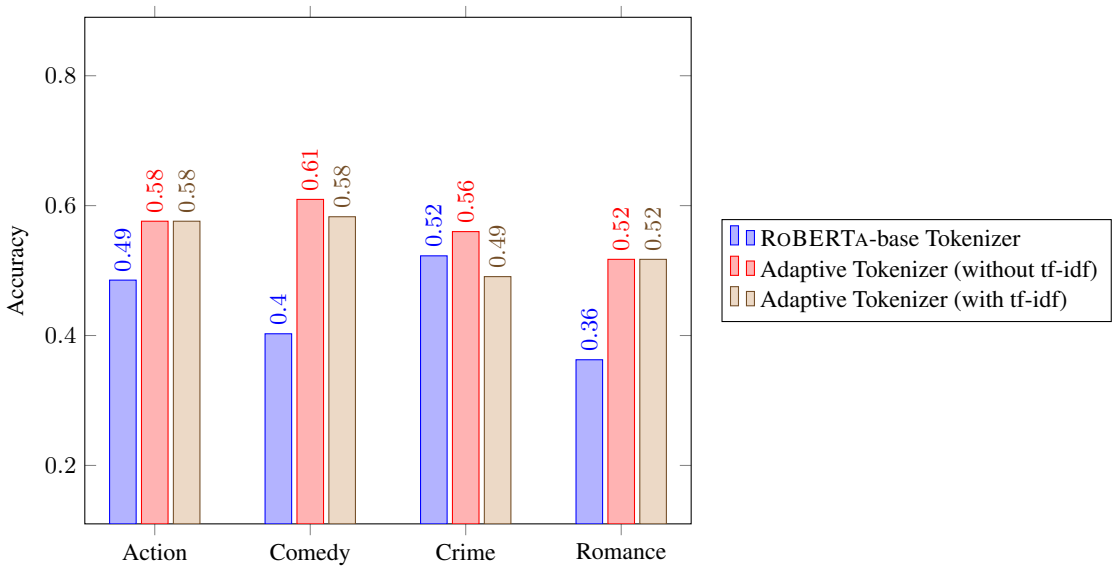


Figure 5: Comparison of accuracy between RoBERTa-base and Adaptive Tokenizer (without and with tf-idf) for action, comedy, crime, and romance genre.

**F1-score** As depicted in Figure 6, a similar trend was observed in the F1 scores. The Adaptive Tokenizer (without tf-idf) generally provided the highest boost in performance. The F1 score for the comedy genre saw an increase from 0.53 with the base tokenizer to 0.56 with AT. However, in the crime genre, AT with tf-idf achieved a higher score (0.46) compared to AT without tf-idf (0.43), indicating a nuanced impact of tf-idf in different genres.

The results demonstrate that AT, especially without tf-idf, significantly enhances both accuracy and F1 score across most genres compared to the baseline RoBERTa-base tokenizer. Specifically, using AT (without tf-idf) led to a substantial 27.44% increase in overall prediction accuracy and a marginal 1.2% increase in the F1-score. In contrast, employing AT (with tf-idf) resulted in a 22.17% improvement in accuracy but a slight decrease of 0.59% in the F1-score. These outcomes suggest that the adaptive tokenizer has a more noticeable impact on accuracy than on the F1-score, which can be ascribed to accuracy's inclusion of true negatives (accurately identified unhelpful reviews), thereby offering a more comprehensive performance assessment. Consequently, accuracy is preferred over F1-score as the more appropriate metric for our dataset's characteristics. However, the addition of tf-idf to AT presented mixed results, indicating that its effectiveness might vary by genre. A more definitive conclusion is hindered by limitations in computational resources, which restrict experimentation with different balances between KL-divergence and tf-idf.

## 5 Discussion

### 5.1 Adaptive Tokenization Architecture

Our exploration into incorporating tf-idf with Adaptive Tokenization led to somewhat mixed outcomes. The effectiveness of tf-idf was less than expected, partly due to the unique nature of our dataset. Movie reviews often include informal language and slang, like "sus," which don't align well with tf-idf's strengths in emphasizing more distinctive words. Furthermore, many movies span multiple genres, leading to an overlap in key token sequences, for instance between action and crime genres. This overlap reduces the ability of tf-idf to distinctly enhance genre-specific tokenization, as

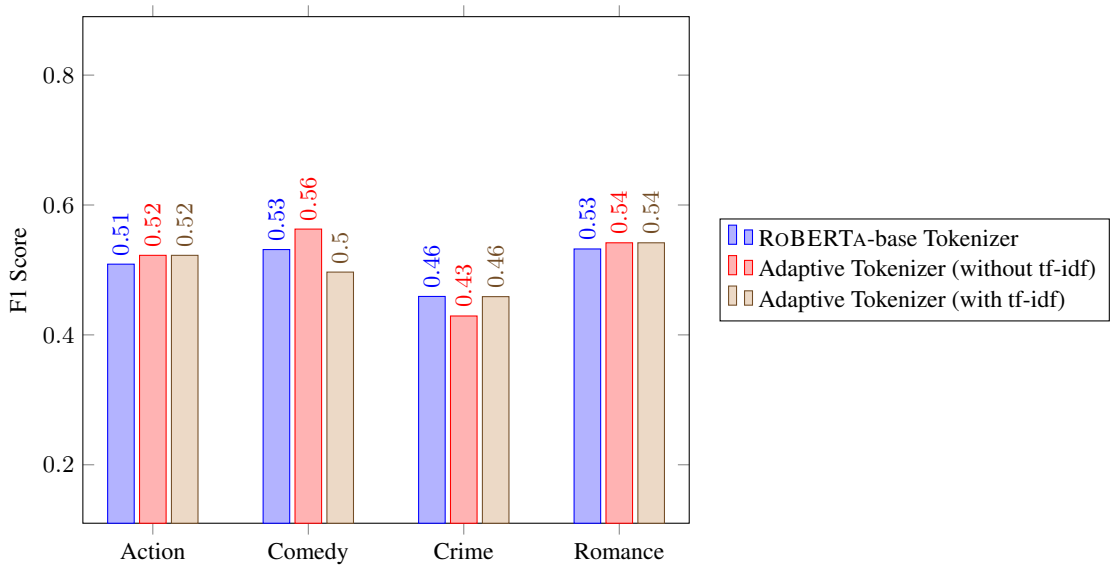


Figure 6: Comparison of F1-score between RoBERTa-base and Adaptive Tokenizer (without and with tf-idf) for action, comedy, crime, and romance genre.

the differentiation in vocabulary isn't as clear-cut.

As described in Section 4.1, introducing extra hyperparameters at the initial stage of our process led to some improvement, although the underlying reasons for this remain somewhat unclear. In terms of using the `softmax` function to compute token sequence distribution, we conjecture that its effectiveness may stem from the fact that it ensures the probabilities sum up to 1. In contrast, the empirical distribution does not necessarily maintain this property after the removal of sequences appearing less than 20 times in the corpus (as outlined in line 11, Algorithm 1). This distinction could account for the marginal improvements we observed, indicating that the method of computing probability distributions can subtly impact the tokenization process's efficacy."

The overall limited enhancement in performance can be attributed to the similarity between the movie review corpus and the base corpus (book corpus), which includes fiction. The vocabulary and linguistic structures in movie reviews are not substantially different from those found in fictional literature. Therefore, the task wasn't as much an 'out-of-domain' challenge as initially anticipated, which might explain why the improvements, though present, were not as significant as expected in a more divergent domain adaptation scenario.

### 5.2 Future Directions

**Enhancement in Augmentation Techniques** Our current approach to subword-based initialization hinges on averaging the embeddings of tokens within select sequences.

This could potentially lead to inaccuracies during model fine-tuning if the subwords don't reflect the word's domain-specific usage. A possible future direction is to employ a **projection-based initialization** approach as in Sachidananda et al., 2021. This involves creating a mapping from static token embeddings to the context-aware embedding space using stochastic gradient descent. Such a map would align RoBERTa's initial token embeddings with word2vec embeddings trained on general and domain-specific texts, potentially improving the model's performance with domain-specific vocabulary.

Further exploration could also include an attention-based mapping strategy. This method would harness the self-attention mechanism to craft embeddings for new token sequences, offering a more dynamic and contextually informed initialization of tokens added during augmentation. Nevertheless, it must be acknowledged that attention-based mapping may entail considerable computational costs. Identifying an equilibrium between the computational demands and the efficacy of domain adaptation poses a significant challenge.

**Project Extensions** The focus of this project has been narrowed to genre adaptation, with the aim of shedding light on broader domain adaptation strategies. Consequently, the enhancement in domain adaptation was not markedly pronounced. There-

fore, potential future extensions of this work may entail exploring actual domain adaptation by developing adaptive tokenizers for a variety of fields such as medical, legal, or fiction, utilizing diverse datasets potentially sourced from repositories such as arXiv. It is also important to note that NLP tasks might vary across different domains, necessitating tailored approaches.

Although this project was constrained by the capabilities of the RoBERTa model, which is not inherently suited for text generation, we are intrigued by the prospect of applying the adaptive tokenization method across other language models and a spectrum of NLP tasks, particularly those involving text generation. The adaptability of tokenization is crucial in these contexts and could play a pivotal role in the success of domain adaptation for NLP models designed for generative tasks.

## Code and README

The code for this project can be found here: `https://drive.google.com/drive/folders/153fw7BcOBXKfbqcn5q0d3okdkpfZbB7V?usp=sharing` (only University of Michigan email can access). The README file is in the folder.

## Acknowledgement

Special thanks to GSI Jake Sansom for discussing our ideas with us, and provided us with Great Lakes resources. We also appreciate John Thiels for his guidance in resolving CUDA and GPU issue in Great Lakes.

## References

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Hady Elsahar and Matthias Gallé. 2019. To annotate or not? predicting performance drop under domain shift. 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)*, pages 2163–2173, Hong Kong, China. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Derivational morphology improves bert's interpretation of complex words.

Jason S Kessler. 2017. Scattertext: a browser-based tool for visualizing how corpora differ. *arXiv preprint arXiv:1703.00565*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Burt L Monroe, Michael P Colaresi, and Kevin M Quinn. 2008. Fightin'words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16(4):372–403.

Pradeep Muthukrishnan, Joshua Gerrish, and Dragomir Radev. 2008. Detecting multiple facets of an event using graph-based unsupervised methods. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 609–616.

Aditya Pal, Abhilash Barigidad, and Abhijit Mustafi. 2020. Imdb movie reviews dataset.

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Paul Rayson, Geoffrey N Leech, and Mary Hodges. 1997. Social differentiation in the use of english vocabulary: some analyses of the conversational component of the british national corpus. *International Journal of Corpus Linguistics*, 2(1):133–152.

Leonard Richardson. 2007. Beautiful soup documentation. *April*.

Vin Sachidananda, Jason Kessler, and Yi-An Lai. 2021. Efficient domain adaptation of language models via adaptive tokenization. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 155–165, Virtual. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.