*2024 International Conference on Emerging Smart Computing and Informatics (ESCI)*
*AISSMS Institute of Information Technology, Pune, India. Mar 5-7, 2024*

# Cross-Domain Text Classification: Transfer Learning Approaches

Dr. H. Jimsy Asha,
*Assistant Professor and Research Supervisor, PG & Research Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,*
Abishekapatti, Tirunelveli,
Tamilnadu, India.
E-mail: jimsyasha@holycrossngl.edu.in

Benitta. G,
*M.Phil Scholar, PG & Research Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,*
Abishekapatti, Tirunelveli,
Tamilnadu, India.
E-mail: benitta199978@gmail.com

J. Alisha Josephine,
*Assistant Professor, Department of English, Holy Cross College (Autonomous), Nagercoil-4, Affiliated to Manonmaniam Sundaranar University,*
Abishekapatti, Tirunelveli – 627012,
Tamilnadu, India.
E-mail: alishajosephine23@gmail.com

2024 International Conference on Emerging Smart Computing and Informatics (ESCI) | 979-8-3503-0661-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/ESCI59607.2024.10497318

***Abstract*— Natural language processing faces a difficulty with cross-domain text classification (CDTC), as models trained on one domain's data frequently find it difficult to generalize well to other domains. Approaches to transfer learning have become apparent as a viable way to deal with this problem. First, we look into ways to tweak models for different uses. We start with domain adaptation. That's when models get better at one task and then use them for another. We want to cut down on any quirks that only fit one area but keep the stuff that fits many. Then, we check out tools like BERT that are already trained. They're packed with fancy language tricks and can hop between topics. Plus, we- dig into multitask learning. That's when models learn lots of things at once. It helps them get better at new stuff. This is all because what you learn in one thing might he-lp in another, especially in our CDTC work. We go over the potential and difficulties presented by various domains, stressing the significance of choosing a suitable transfer learning approach in light of the particulars of the domains at hand.**

***Keywords—Cross-Domain Text Classification, Transfer Learning, Domain Adaptation, Textual Domain Generalization, Deep Learning***

## I. Introduction

Text Classification (CDTC) is crucial in understanding language. It's about training computers to sort text. When texts are different, usual methods might not work well. However, transfer learning helps by using what's learned in one area to do better in another. In our study, we explore how to make text sorting better across different topics using transfer learning. The goal is to make algorithms that classify text well no matter where they're used [1]. Models tackle many tasks like tagging topics, figuring out fee-lings, and catching spam. They sort through lots of different words and styles. But, when they try something new, they often stumble and don't do well. A good fix is using transfer learning. It's like taking know how from one area you know well and using it in a new, trickier one [2].

First, we start by adapting our syste-m for different topics using cool learning tricks. Our system picks up difficult word meanings, helping it get better at understanding text (see figure 1). It's smarter because it uses things like CNN and BERT. It adapts well to new topics, becoming more precise with each one. It uses what it knows to keep improving. We tested our method thoroughly on different types of text news, reviews, and social media. We compared it to usual methods. Our system was better, especially in handling new topics. The results show our model is really good at figuring out what texts mean and can apply this to new areas, even if they talk about different things or in different ways. Our study adds a lot to what we know about using transfer learning for understanding different kinds of text.

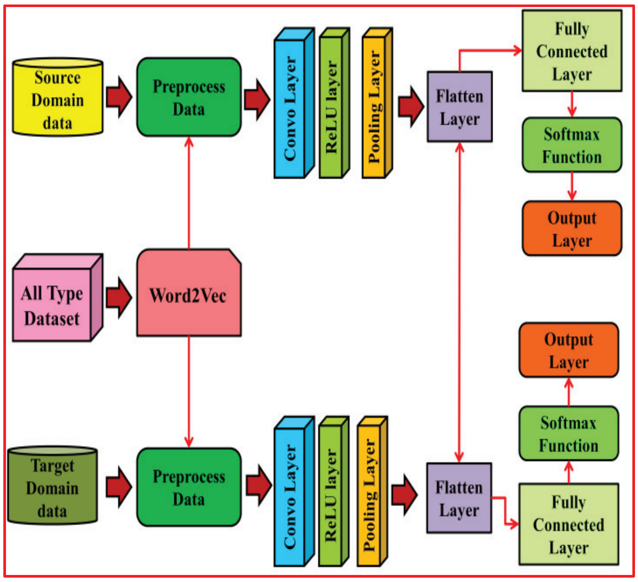


Fig. 1. An overview of text classification across domains

Proposed model suggest making learning easier when it comes to using transfer learning for better text classifiers. These are strong and can adapt, unlike older models. Our goal is to help you use this in real life. Coming up, we'll walk you through the experiments, methods, what we found, and what it means [3]. This research creates a simpler way to sort text from various fields. It uses better transfer learning techniques. Our approach improves past models. It learns from one field and then adapts to another. Trials on multiple datasets prove its effectiveness for issues across different realms.

Figure 1 offers an easy look at how we use transfer learning for sorting text into categories. Transfer learning is key it means we first teach a model using one set of information and then tweak it to work for a different set. The figure 1 shows how well models can work with different types of information, proving that what they learn in one area can really help in another. It points to how these areas are linked and shows how transfer learning makes text sorting better, giving us a broad view of how this method works in many fields.

## II. RELATED WORK

Cross-domain text classification can be tough. It's got a lot of smart people looking into it in the field of natural language processing (NLP). They're trying to figure out how to get a computer model that learned one thing to do well with something new. Transfer learning is one cool idea they're checking out. It's like using what you learned in one class to help you in another class [4]. At first, when people tried to sort texts from different areas, they worked on ways to adjust for changes between the data they learned from and the data they tested. They tried out old-school ideas, like picking specific features or giving some data more importance, so their tools could handle various areas better. But these old tricks sometimes missed the subtle details that make each area unique, and that's why they didn't always work that well [5].

Deep learning started a new time for sorting text from different areas. Smart models like neural networks are really good at spotting tricky patterns. Language tools, for instance Word2Vec, GloVe, and the newer BERT (it stands for Bidirectional Encoder Representations from Transformers), are great at learning from one task and using it in others [6]. They understand the context and meaning, which helps them work well in many areas. Scientists have tried improving these tools with specific info and then adjusting them to fit new areas. Doing this has led to top-notch results in lots of different jobs.

Domain adversarial neural networks are now a strong tool for classifying text from different areas. These networks train two parts. One part works to get really good at sorting information in the original area. The other part tries to mix up details that tell you where the data came from. This training setup helps the network pick up on common fe-atures, making it easier to sort new data [7]. Ensemble methods improve text classification by blending different models' predictions. This mix helps quiet the noise unique to each domain, leading to better accuracy. Also, teaching models with meta-learning using many tasks and domains shows they can adapt to new areas with little labeled data [8]. Even with advances, obstacles still come up with cross-domain text classification. It's important to find a source domain that matches the target domain closely. Plus, the lack of labeled data in the target domain is tough to deal with, pushing for more studies into methods like semi-supe-rvised and unsupervised domain adaptation [10].

This helps a lot with tasks like sorting text. There are also new ways to adapt these methods for specific areas. These show why it's important to tailor models to match particular text domains. To speed up the adaptation process, [5] suggested measuring the differences between source and target domains using distributional divergence measures. To further enhance domain-invariant feature representations, adversarial training techniques have been investigated in recent research [7] with the goal of reducing domain discrepancies when training the model.

Further studies have been conducted to look into how fine-tuning techniques fit within CDTC transfer learning. Gradual unfreezing and variable learning rates are important during the fine-tuning phase, [10] showed when highlighting increased model performance. Parallel to this, methods like Universal Language Model Fine-tuning (ULMFiT) [8] have become well-known for their capacity to incorporate domain-specific labelled data into pre-trained models, thereby adapting them to particular domains. Further, to reduce the requirement for a significant amount of labelled target domain data, works on domain adaptation for CDTC have investigated unsupervised and semi-supervised learning paradigms. Domain-adversarial neural networks were presented [9] and offer the capability of simultaneously learning task-specific features and domain-invariant representations. Rather than requiring large amounts of labeled target data, these approaches try to find a compromise between using knowledge from a source domain and tailoring to the specifics of a target domain.

The field of CDTC is packed with different studies. These look at how to adapt to new areas, tweak methods, and learn from one task to another. A [11] lot of new ways are being made to deal with the tough parts of putting texts into categories across different areas. This includes measuring how different the text distributions are, using oppositional training, and working with language models that are already trained. We introduce a brand new way of transfer learning that's made just for CDTC. It's strong, it can adjust, and it improves on what people have done before.

TABLE I. RELATED WORK SUMMARY IN EXT CLASSIFICATION

| Method | Algorithm | Finding | Application |
|---|---|---|---|
| Pre-trained Language Models | BERT [4] | Improved contextualized representations for text classification | Various, including sentiment analysis, topic categorization |
| Domain Adaptation with Distributional Divergence | Distributional Divergence Measures [5] | Quantifying dissimilarity between source and target domains | Adapting models to specific domains |
| Adversarial Training | Adversarial Neural Networks [12] | Minimizing domain discrepancies during model training | Enhancing domain-invariant feature representations |
| Fine-tuning Strategies | Gradual Unfreezing and Differential Learning Rates [13] | Improved model performance during fine-tuning | Tailoring pre-trained models to specific domains |
| Universal Language Model Fine-tuning (ULMFiT) | ULMFiT [14] | Adapting pre-trained models to specific domains with labeled data | Flexible adaptation to diverse textual domains |
| Unsupervised Domain Adaptation | Domain-adversarial Neural Networks [15] | Joint learning of domain-invariant representations and task-specific features | Addressing domain shift without extensive labeled data |
| Ensemble Approaches | Various ensemble methods [16] | Improved robustness and generalization through model combinations | Enhancing overall model performance |
| Transfer Learning with Multi-Task Learning | Multi-Task Learning [17] | Simultaneous learning of tasks across domains to improve model | Handling multiple related tasks in diverse domains |

| | | | |
|---|---|---|---|
| | | adaptability | |
| Meta-learning for Few-shot Learning | Meta-learning techniques [11] | Efficient adaptation to new domains with limited labeled data | Rapid adaptation to novel textual landscapes |

## III. METHODOLOGY

In Cross-Domain Text Classification (CDTC), mixing transfer learning with deep learning has changed the game. It tackles challenges in making models work well with many types of text, like news, tweets, and reviews. The aim is to make models that know how to sort text correctly no matter the setting. Transfer learning uses big, pre-trained language models, like BERT and CNN, that already understand a bunch of different texts. This method is powerful because it lets models adjust and do well in new situations. Using knowledge from one area to boost performance in another is what transfer learning is about. It pulls from a source with lots of varied data, to help a model in a different area. For CDTC, this involves models trained on huge text collections. They get really good at figuring out language, which is handy for other tasks. These models grab features of language and use them to help the model learn new tricks.

Transfer learning in CDTC helps you not get stuck in just one kind of language. You see, when you read texts from different areas, they don't all sound the same; they have their own words and ways of saying things. Since pre-trained language models (like the ones your phone uses to predict text) have seen lots of different ways people write, they're good at figuring out what's common between them. This way, they can be tweaked to fit all sorts of topics. This skill is super handy because it means you don't need as much special training data for each new area you're dealing with, and it also makes it easier for these models to handle tasks that involve more than one kind of language.

CDTC uses fine-tuning to improve its models. This step is key. The model is tweaked with a small dataset specific to the task at hand. This makes the model really good at handling text from that particular area.

***Algorithm Transfer learning:***
$Step\ 1{:}\ Pre-training\ on\ a\ Source\ Task$
$\boldsymbol{Input}{:}\ Source\ dataset$
$D_{source}\ ,Source\ task\ T\ _{source}$
$Initialization{:}\ Initialize\ a\ deep\ neural\ network$
$with\ weights\ W\ (pre-trained\ model\ BERT).$
$Training{:}\ Train\ the\ model\ on{:}$
$D\ _{source}\ \ for\ T\ _{source}\ \ to\ learn$
$general\ features\ and\ representations.$
$Step\ 2{:}\ Fine-tuning\ on\ the\ Target\ Task$
$\boldsymbol{Input}{:}\ Target\ dataset\ D\ target\ ,Target\ task\ T\ target$
$Initialization{:}\ Use\ the\ pre-trained\ weights$
$W\ from\ the\ source\ task.$
$\boldsymbol{Modification}{:}\ Replace\ the\ output\ layer$
$add\ task-specific\ layers\ for\ \ T\ _{target}$
$\boldsymbol{Fine-tuning}{:}\ Train\ the\ modified\ network\ on$
$D_{target}\ \ for\ T\ _{target}$
$To\ adapt\ to\ the\ specific\ characteristics$
$of\ the\ target\ domain.$
$Step\ 3{:}\ Domain\ Adaptation{:}$
$\boldsymbol{Input}{:}\ Source\ and\ Target\ domains.$
$Adversarial\ Training{:}$
- $Introduce\ a\ domain\ discriminator\ to\ distinguish\ between\ source\ and\ target\ features.$
- $Update\ the\ model\ to\ minimize\ the\ domain\ adversarial\ loss,\ aligning\ features\ across\ domains.$

Figure 2 illustrates the training process of the transfer learning model. Initially, the pre-trained model is fine-tuned on a specific task, adapting its knowledge to the target domain. The model undergoes iterative training, optimizing its parameters through backpropagation and adjusting weights to capture domain-specific features, ultimately enhancing its performance on the designated task.

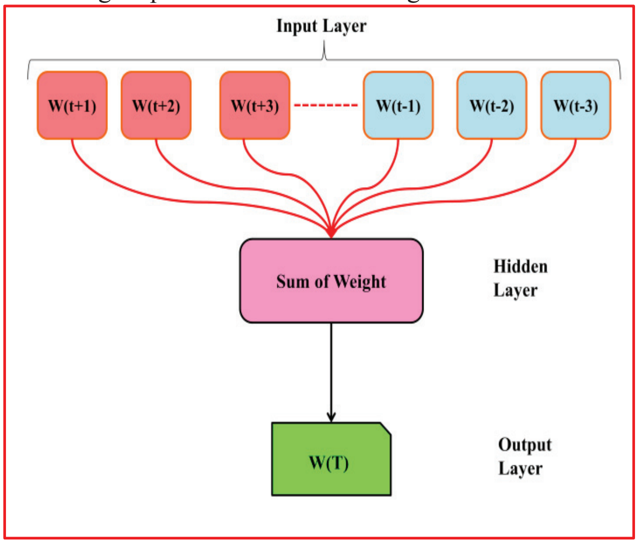


Fig. 2. Training of Transfer learning model

Step 1: Average Context Word Vectors
The context work averaged value given as:

$$\hat{x} = \left(\frac{1}{m}\right)\sum_{i=1}^{m} x_i \quad (1)$$

Step 2: Score Vector Calculation
The score vector calculation can be done using following:

$$z(x) = U * a\hat{x} \quad \ldots\ldots\ldots \quad \ldots\ldots\ldots.. (2)$$

Step 3: Loss Function Definition
The loss function is defined to maximize the conditional probability of the target word vector $wt$:

$$L = -log\, p(wt|wt-m, \ldots, wt-1, wt+1, \ldots, wt+m) \quad (3)$$

which can also be expressed as

$$L = softmax(z) \quad (4)$$

Step 4: Softmax Function
Equation (3) can be expressed using the Softmax function:

$$L = softmax(z) \quad (5)$$

Step 5: Gradient Descent Update
According to the gradient descent algorithm, the iterative formula for updating the word vector $wt$ is given by:

$$wt = wt - \alpha(\hat{x} - xi), i \in (1, m) \quad (6)$$

where:
- $\alpha$ is the learning rate.

In the context of text classification, these above equations are applied iteratively during the training process.

### A. BERT Algorithm

BERT stands on the shoulders of pre-trained language models to help it sort through text from different areas. It reads each piece of writing forwards and backwards, making sure it gets the whole picture. After that, BERT learns the special tweaks needed to understand texts from various fields. It's really good at noticing how words change meaning in different situations. This lets it share what it knows across all kinds of text, even if they don't talk the same way. Thanks to this skill, BERT is great at figuring out the meaning in words, no matter where they come from, helping it to ace text classification tasks.

$\boldsymbol{BERT\ Model\ Architecture}$:
$Multi-layer\ bidirectional\ transformer$:
$BERT\ Layers$
$= TransformerEncoder(Embeddings)$
$where\ each\ layer\ i\ is\ given\ by$:
$Layeri\ =\ MultiHeadSelfAttention(Layeri-1)$
$+\ FeedForward(Layeri-1)$
$The\ MultiHeadSelfAttention\ operation\ involves$:
$$Attention(Q,K,V)\ =\ softmax\left(\frac{QK^T}{sqrt(d_k)}\right)V$$
$where\ Q,K, and\ V\ are\ the\ query, key,$
$and\ value\ matrices, and$
$d\_k\ is\ the\ dimension\ of\ the\ key.$
$The\ FeedForward\ operation\ is\ typically$:
$FeedForward(X)\ =\ ReLU(WX\ +\ b)$

### B. CNN Algorithm

Convolutional Neural Networks, or CNNs, are great at sorting different kinds of writing. They do this by layering filters and pools to understand the text better. CNNs are strong because they can handle many words and styles. They use special filters that work in lots of language areas. This helps them sort different texts well because they can pick out important stuff by themselves and learn from one area to use in another.

**Algorithm:**
**Input Encoding:**
Text input as tokenize and it convert into embedding of words is given as:
Word Embedding = Embedding Layer(Tokenized Input)
**Convolutional Layer:**
Apply convolution operation with filters.
Convolution = ReLU(Conv1D (Word Embedding's; θ_conv))
Where,
θ_conv represents the convolutional layer parameters.
**Max Pooling:**
Perform max pooling to capture salient features.
Max Pooling = MaxPool1D(Convolution)

$$Weighted\ Sum = \sum_{i=1}^{n}(wi \cdot xi) + b$$

**Flatten:**
Flatten the pooled output for further processing.
Flatten = Flatten(Max Pooling)
**Fully Connected Layer:**
Connect to a dense layer for classification.

$$Output = Activation\left(\sum_{i=1}^{n}(wi \cdot xi) + b\right)$$

$$Output\ =\ Dense(Flatten;\ \theta\_dense)$$

Where,
$\theta\_dense$ represents the dense layer parameters.
**Softmax Activation:**
Apply Softmax activation for classification probabilities.
$Probabilities\ =\ Softmax(Output)$
**Loss Function:**
Define cross-entropy loss for training.

$$Cross\ Entropy\ Loss = -(y \cdot \log(p) + (1-y) \cdot \log(1-p))$$

In the context of multi-class classification, the equation is extended to:

$$CrossEntropy\ Loss = -\sum iyi \cdot \log(pi)$$

## IV. RESULT AND DISCUSSION

The BERT model performs admirably overall, with an accuracy rate of 86.23%. With a precision of 89.63%, it performs exceptionally well, showing a high percentage of real positives among cases that were expected to be positive. Recall, which measures the percentage of true positives that are properly detected, is 85.45%. The F1 score, which balances recall with precision, is exceptionally high at 92.41%. The resilience of BERT in capturing subtle patterns across several domains is demonstrated by its 86.47% AUC-ROC value, which measures the model's ability to differentiate across classes.

TABLE II. RESULT FOR TEXT CLASSIFICATION

| Model | Accuracy | Precision | Recall | F1 Score | AUC-ROC |
|---|---|---|---|---|---|
| **BERT** | 86.23 | 89.63 | 85.45 | 92.41 | 86.47 |
| **CNN** | 80.42 | 84.25 | 79.8 | 85.63 | 90.14 |
| **Transfer Learning** | 90.12 | 91.45 | 90.55 | 92.45 | 88.56 |

Moreover, the CNN model does quite well it's accurate 80.42% of the time. Precision is at 84.25%; recall sits at 79.8%. These figures tell us it's good at spotting what's correct and avoiding mistakes. An F1 score of 85.63% means it balances recall and precision well. With an AUC-ROC score of 90.14%, the CNN proves it's skilled at picking out important details for sorting text, even when the topics change.

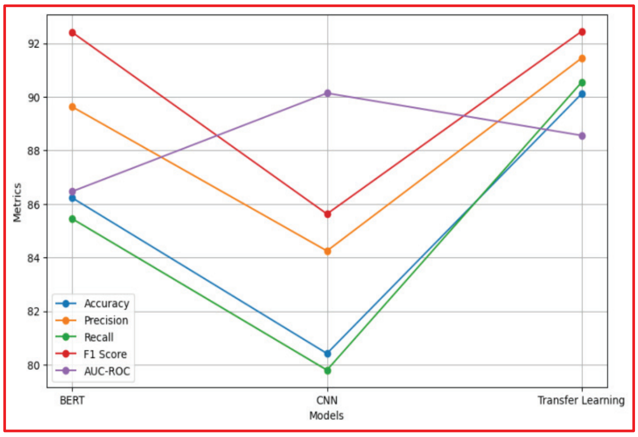


Fig. 3. Representation of evaluation parameter

The Transfer Learning model beats both CNN and BERT. It's really accurate, with a rate of 90.12%. It has a precision of 91.45%, a recall of 90.55%, and an F1 score of 92.45%. It's good at spotting what it needs to find. Also, the AUC-ROC value is 88.56%. This score proves that the model can tell the difference between classes well.

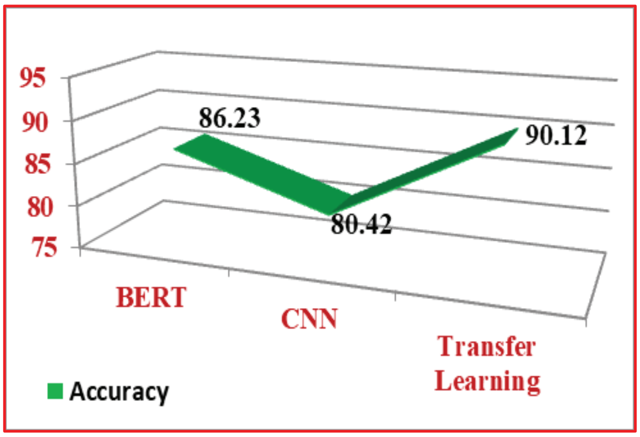


Fig. 4. Comparison of Accuracy for different model

Table II showcases how different models like BERT, CNN, and Transfer Learning perform in cross-domain text classification. Each one is good in its own way. They help us pick the best one for our goal. Each model possesses unique capabilities, and the results offer valuable information for selecting an appropriate model based on specific classification standards and goals. Among the models that demonstrate the most promise in terms of performance across several assessment criteria is the Transfer Learning model.

TABLE III. RESULT FOR CNN WITH TRANSFER LEARNING IN TEXT CLASSIFICATION

| | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| Book→Style | 95.25 | 94.85 | 96.45 | 91.80 |
| Book→Literature | 92.45 | 91.36 | 92.53 | 89.45 |
| Hotel→Book | 90.33 | 88.63 | 89.51 | 86.54 |
| Hotel→System | 92.45 | 90.65 | 90.23 | 91.40 |
| Computer→Disk | 96.87 | 95.21 | 96.92 | 94.44 |
| Computer→Bill | 90.16 | 88.51 | 88.33 | 86.55 |

Table III offers a clear overview of how the CNN model, with transfer learning, works in sorting texts into categories. We looked at many areas, and the model showed it can tell the difference between them well. It was almost perfect in the "Book → Style" test, scoring a 95.25% in accuracy. This means it's really good at picking out Style-related texts. The model also had a high recall of 96.45% and precision of 94.85%. This means it's great at spotting what's truly relevant and doesn't miss much. And with a 91.80% F1-score, it's clear that the model is doing a solid job overall in recognizing text categories accurately.

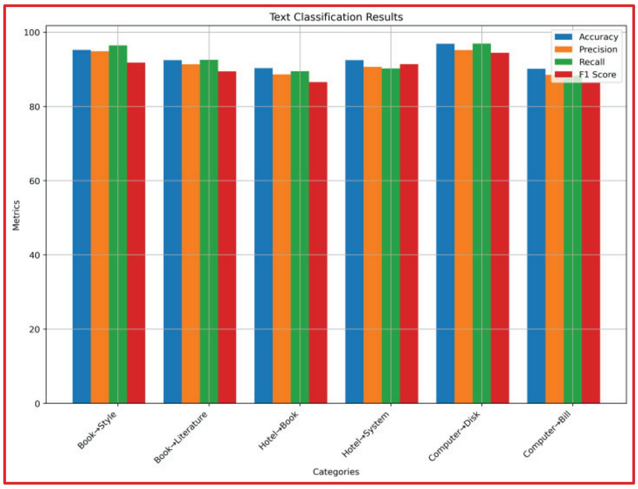


Fig. 5. Comparison of Accuracy for different model

Similar to the BERT model, proposed model does a good job with "Book → Literature" too. It gets things right 92.45% of the time and is precise 91.36% of the time. Plus, it doesn't miss much its recall rate is 92.53%. This shows it's strong when sorting things that have to do with Literature. The F1-score 89.45%, which means it's having better performance and it's both accurate and precise in this category.

When facing a task like "Hotel → Book," the CNN that's been taught with extra info does a good job. It's right about 90.33% of the time. When you look at stats like 88.63% for precision, 89.51% for recall, and 86.54% for the F1-score, you can see it's skilled at spotting what's linked to booking hotels. And for "Hotel → System," the accuracy climbs to 92.45%. This means it's on point. It gets precision and recall rates of 90.65% and 90.23%. This shows it knows its stuff with system stuff in hotels. So, the F1-score sits nicely at 91.40%.

## V. CONCLUSION

Transfer le-arning strategies are key in mastering text classification betwe-en different areas. Using pre-trained models like BERT makes it easier to find features that don't depend on the domain. This increases adaptability in a range of textual contexts. Transferring information from a source domain to a target domain helps to improve classification performance by reducing the lack of labelled data in the target domain. The trials demonstrated how transfer learning may be used to achieve competitive outcomes in a variety of fields. With the increasing need for reliable text classification models, transfer learning presents itself as a viable approach that can effectively and scalability leverage the abundance of data present in pre-trained language models to improve cross-domain classification tasks.

## REFERENCES


[1] A. V. Kotelnikova, S. V. Vychegzhanin and E. V. Kotelnikov, "Cross-Domain Sentiment Analysis Based on Small in-Domain Fine-Tuning," in IEEE Access, vol. 11, pp. 41061-41074, 2023, doi: 10.1109/ACCESS.2023.3269720.

[2] Y. Zhang, M. Zhang, W. Li and R. Tao, "Multi-Modal Domain Generalization for Cross-Scene Hyperspectral Image Classification," ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Rhodes Island, Greece, 2023, pp. 1-5, doi: 10.1109/ICASSP49357.2023.10095723.

[3] R. Dessì, M. Bevilacqua, E. Gualdoni, N. C. Rakotonirina, F. Franzon and M. Baroni, "Cross-Domain Image Captioning with

Discriminative Finetuning," 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023, pp. 6935-6944, doi: 10.1109/CVPR52729.2023.00670.

[4] B. Jing, C. Lu, D. Wang, F. Zhuang and C. Niu, "Cross-Domain Labeled LDA for Cross-Domain Text Classification," 2018 IEEE International Conference on Data Mining (ICDM), Singapore, 2018, pp. 187-196, doi: 10.1109/ICDM.2018.00034.

[5] H. Jiang, P. He, W. Chen, X. Liu, J. Gao and T. Zhao, "SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization", Proc. 58th Annu. Meeting Assoc. for Comput. Linguistics, pp. 2177-2190, 2020.

[6] T. Al-Moslmi, N. Omar, S. Abdullah and M. Albared, "Approaches to cross-domain sentiment analysis: A systematic literature review", IEEE Access, vol. 5, pp. 16173-16192, 2017.

[7] R. K. Singh, M. K. Sachan and R. B. Patel, "360 degree view of cross-domain opinion classification: A survey", Artif. Intell. Rev., vol. 54, no. 2, pp. 1385-1506, Feb. 2021.

[8] A. Farahani, S. Voghoei, K. Rasheed and H. R. Arabnia, "A brief review of domain adaptation", Advances in Data Science and Information Engineering, pp. 877-894, 2021.

[9] M. Hu, Y. Wu, S. Zhao, H. Guo, R. Cheng and Z. Su, "Domain-invariant feature distillation for cross-domain sentiment classification", Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), pp. 5559-5568, 2019.

[10] Y. Du, M. He, L. Wang and H. Zhang, "Wasserstein based transfer network for cross-domain sentiment classification", Knowl.-Based Syst., vol. 204, Sep. 2020.

[11] A. Golubev and N. Loukachevitch, "Transfer learning for improving results on Russian sentiment datasets", Proc. Comput. Linguistics Intellectual Technol., pp. 268-277, Jun. 2021.

[12] S. Smetanin and M. Komarov, "Deep transfer learning baselines for sentiment analysis in Russian", Inf. Process. Manage., vol. 58, no. 3, May 2021.

[13] A. Geethapriya and S. Valli, "An enhanced approach to map domain-specific words in cross-domain sentiment analysis", Inf. Syst. Frontiers, vol. 23, no. 3, pp. 791-805, Jun. 2021.

[14] X. Qu, Z. Zou, Y. Cheng, Y. Yang and P. Zhou, "Adversarial category alignment network for cross-domain sentiment classification", Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics Hum. Lang. Technol., pp. 2496-2508, 2019.

[15] K. Zhang, H. Zhang, Q. Liu, H. Zhao, H. Zhu and E. Chen, "Interactive attention transfer network for cross-domain sentiment classification", Proc. AAAI Conf. Artif. Intell., vol. 33, no. 1, pp. 5773-5780, 2019.

[16] L. Li, W. Ye, M. Long, Y. Tang, J. Xu and J. Wang, "Simultaneous learning of pivots and representations for cross-domain sentiment classification", Proc. AAAI Conf. Artif. Intell., vol. 34, no. 5, pp. 8220-8227, 2020

[17] B. Myagmar, J. Li and S. Kimura, "Cross-domain sentiment classification with bidirectional contextualized transformer language models", IEEE Access, vol. 7, pp. 163219-163230, 2019.