

Text grouping: a comprehensive guide

Padarabinda Palai¹, Kaushiki Agrawal¹, Debani Prasad Mishra², Surender Reddy Salkuti³

¹Department of Computer Science Engineering, IIIT Bhubaneswar, Odisha, India

²Department of Electrical Engineering, IIIT Bhubaneswar, Odisha, India

³Department of Railroad and Electrical Engineering, Woosong University, Daejeon, Republic of Korea

Article Info

Article history:

Received Jan 16, 2022

Revised Jul 3, 2022

Accepted Jul 16, 2022

Keywords:

Neural networks

Natural language processing

Query categorisation

Term frequency-inverse

document frequency

Transformers

ABSTRACT

Text keywords have huge variance and to bridge the gap between the country business segment which provides negligible information and the keywords that have a huge longtail it is imperative for us to categorize the queries that provide a middle ground and also serve a few other purposes. The paper will present those in-depth. Query categorization falls into the segment of 'Multi-Class Classification' in the domain of natural language processing (NLP). However, business requirements require the implementation of any technique that could provide as accurate results as possible. So, to solve this problem the paper discusses an amalgamation of approaches like TF-IDF (term frequency-inverse document frequency), neural networks, cosine similarity, transformers-all of which fix specific issues.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Surender Reddy Salkuti

Department of Railroad and Electrical Engineering, Woosong University

Jayang-dong, Dong-gu, Daejeon-34606, Republic of Korea

Email: surender@wsu.ac.kr

1. INTRODUCTION

In large-scale organizations, analytics is a major domain of research. There are different aspects to data analytics though [1]. They are mainly descriptive, predictive, or prescriptive in nature. However, the final aim remains the same - to prevent losses and to gain profits. Now, all that computation of numbers that takes place has to be visualized in some manner [2]. The more intuitive the representation is the more accurate the subsequent action is. Power BI happens to be one such tool that makes the visualization of complex data seamless [3]. It is a suite of business analytics tools that are used to give insights. It's intelligent and powerful and hence used widely by multi-national corporations (MNCs) to generate reports and dashboards [4]. My research required me to spend some time understanding and working on ways to visualize the different metrics of user search queries.

Text grouping enables product and business teams in strategic business making to improve engagement on the platform and provides visibility before/after launch/fix [5]. The idea is to develop a descriptive model to analyze texts by selecting content management system (CMS) combination along with the volume of texts, click through rate (CTR%), and average (clicked) result position [6]. Furthermore, attempt to develop a language processing model for the categorization of queries in buckets for quick identification of underperforming search queries in terms of their respective categories [7]. Without such a solution in place stakeholders of businesses aren't in a position to derive any information whatsoever, about the user search habits and consequently are deprived of the chance to make progressive business decisions that would drive profit [8]. Without a categorization model in place, it would be immensely laborious to get any meaningful information from the raw search queries simply because they have a huge variance [9].

The problem that businesses face right now is, they can't figure out where the change in search performance is coming from [10]. It's imperative to understand the trend in text frequency metrics. If there is a query categorisation in place not only can the areas of weakness be identified, but also a sustainable model that could be built to work on under-performing categories and help business stakeholders make informed decisions. The paper will present the need for query categorization in some time.

Let's assume that a company's search engine optimization (SEO) team tracks user queries along with additional search metrics like the number of searches for the respective queries, the CTRs, and so on. But when you think about it, only a handful of user queries will be an exact match with their business product numbers. But what if there is a typing error? What if a product named 'GTX 1050ti' is incorrectly queried as 'GXT 1050ti' or 'Alienware' as 'Aleinware'? The search engine indexes it and omniture (or any other search analytical tool) captures it as a different search query. But the intent is the same, which is a laptop/graphics card for example. This is just one of the gaps existing with the current analysis workflow. More importantly, it's impossible to get a lot of insights from raw user queries because there is a huge variance. They can be literally anything. Even products irrelevant to the business.

2. RESEARCH METHOD

2.1. The problem

Develop a descriptive model to process and accurately group user queries into high-level categories that would aid in identifying and gauging the volume and performance of texts. Findings would help the engineering, development, and business stakeholders to improve understanding of the keyword usages as key performance indicator (KPI) drill through capability and fine-tune lower performance query categories [11]. Currently, businesses don't have the ability to explain changes referring to searches. Different domains of businesses, receive millions of searches. Searches have a very large variance. They can be anything. Businesses can't keep track of keyword-level information. It's impossible to understand how the dynamic is changing w.r.t country level and in a specific time period. Lack of service that can bridge the gap in the root cause analysis (RCA) step.

2.2. Need for query categorisation

Query categorization provides a solution by grouping the keyword level KPIs or keyword category level [12]. Right now when you refer to any search visualization report, analysts generally start analyzing text search performance on a country level [13]. Why? because different countries have different users who have different user habits. So typically, they start on a country level and then drill down to the country segment level viz also referred to as the business segment level. Large businesses usually have multiple business segments and on different levels [14]. Like business segments, consumer segments of business to business (B2B) segments.

Query categorisation provides a middle ground between country/business segment level and keyword level. The query categorization bridges the gap and makes it simple to derive the mix and engagement of high performance and low-performance categories within a country [15]. Without this, there is no ability to explain the origin of changes.

- Tracking of the volume of demand: With such a solution in place it could be understood what keyword category causes issues and subsequent abilities to zoom in to more specific areas. In general, analysts would have granular control over the volume and demands of the searches on the domains of the organization [16].
- Keyword level information: With query categorization in place, keyword-level information could be managed efficiently [17]. It could not only classify queries in single categories but subcategories as well. So that could help us gain insights into a hierarchical structure. Both top-down and bottom-up approaches [18].

2.3. Categorization models

This is a classical problem of text classification. Since there are categories/classes involved, it becomes a multi-text classification problem. Text classification otherwise called text labeling is the way toward sorting text into coordinated gatherings [19], [20]. By utilizing natural language processing (NLP), text classifiers can consequently investigate text and afterward allocate a bunch of pre-characterized labels or classifications dependent on its substance.

2.3.1. Categories

Let's assume a company, 'Bizon' has 4 major domains as had been briefly mentioned in the introduction. They are 'Bizon.com' for business to consumer (B2C) business, 'supportbizon.com' for support related business, 'bizonpremium.com' for B2B business, and 'bizoncloud.in' as a cloud computing

platform. If one surfs through the entire portfolio of Bizon's websites, one will identify 7 major types of products/services that it offers which are: i) Bizon products: major hardware products like laptops and servers; ii) Bizon software: software licenses, keyboard, headsets, and mouse; iii) Bizon cloud: cloud-based services; iv) Bizon amenity: warranty extension, infrastructure nodes, and consulting; v) Bizon quickfix: self-service support like repair, customer support; vi) Bizon sale: various deals that Bizon offers on its products; and vii) Bizon consultancy: for advisory purposes. So, the primary objective is to categorize all the queries into one of these 7 classes. If there is something completely irrelevant, the plan is to categorize them into another class/bucket [21]. But that's more towards the end and comes under the error feedback mechanism. Obviously, all these categories have their own sub-categories and will be discussed later on in the paper. But now, let's proceed towards the various solutions.

2.3.2. Step 1: TF-IDF vectorizer

The point of term frequency-inverse document frequency (TF-IDF) is to characterize the significance of a watchword or expression inside a record or a web page and can add ngram work as an analyzer in the TF-IDF vectorizer which created the succession of every section then a meager grid is produced, which is utilized to discover the percent closeness of the strings. These calculations as a rule manage numbers, and regular language is, indeed, text and it changes that text into numbers. Vectorizing an archive is taking the content and making one of these vectors, and the quantities of the vectors in some way or another address the substance of the content [22].

```
vectorizer = TV(dataframe=1, analyzer=ngram)
tfidf = vectorizer.fit_transform(new_data_2) #catalogue
NN = NearestNeighbors(n_neighbors=1, n_jobs=-1).fit(tfidf)

def getNearestNeigh(query):
    query_1= vectorizer.transform(query)
    distances, indices = NN.kneighbors(query_1)
    return distances, indices
```

2.3.3. Step 2: LSTMs and GloVe

Recurrent neural networks (RNN's) experience difficulties with transient memory. On the off chance that an arrangement is sufficiently long, they struggle conveying data from prior time steps to later ones. So, in the event that you are attempting to handle a section of text to do forecasts, RNN's may leave out significant data all along [23]. Subsequently, this causes the need for long short-term memory (LSTM) which is an extraordinary sort of RNN's, equipped for learning long haul conditions [24]. LSTM has abilities to recall the data for a significant stretch of time. Here's the model that was formulated:

Model: "sequential_n"

Layer (type)	Output Shape	Param #
embedding_6 (Embedding)	(None, 50, 300)	1891500
spatial_dropout1d_3 (Spatial)	(None, 50, 300)	0
lstm_4 (LSTM)	(None, 100)	160400
dense_4 (Dense)	(None, 7)	707
Total params: 2,052,607		
Trainable params: 2,052,607		
Non-trainable params: 0		

None

GloVe embeddings are vector representations of words and they have been trained over 300 billion unique tokens in the vocabulary [25]. So, I have used GloVe embeddings as the embeddings initializer in the LSTM model. The embeddings matrix comprises all the tokens in the dictionary represented in GloVe embeddings [26]. So, for say there is a term as 'Alienware', the matrix has a row for that word in vector representation but that vector representation is taken from the huge library of GloVe. This is to make the model more accurate. The activation functions and loss functions are mentioned respectively to suit the given context.

2.3.4. Step 3: bidirectional encoder representations from transformers

Bidirectional encoder representations from transformers (BERT) for short is a very popular NLP model from Google known for producing state-of-the-art results in a wide variety of NLP tasks [27]. As shown in Figure 1, BERT models require graphics processing units (GPUs) for faster processing. BERT requires the data to be encoded in a specific format. BERT is fundamentally an encoder pile of transformer design. A transformer design is an encoder-decoder network that utilizes self-consideration on the encoder side and consideration on the decoder side [28].

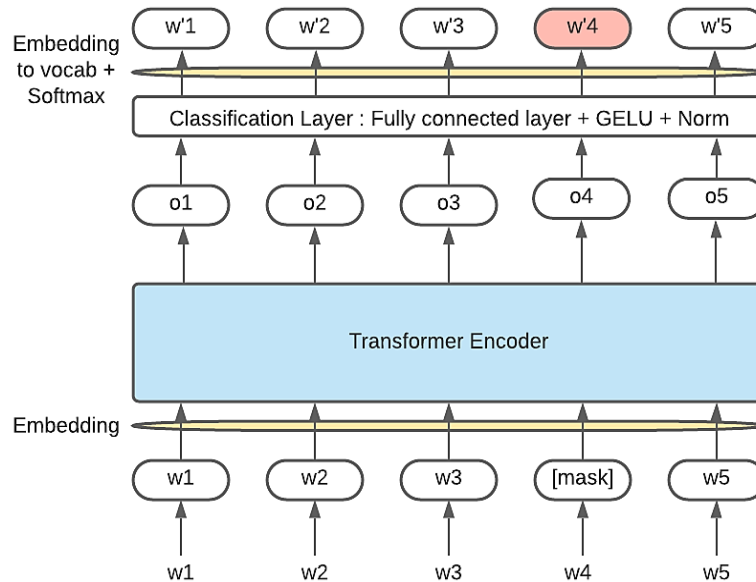


Figure 1. BERT architecture

BERT-base has 12 layers in the encoder stack while BERT-large has 24 layers. These are more than the transformer design portrayed in the first paper (6 encoder layers). BERT models (base and large) additionally have bigger feedforward networks (768 and 1,024 secret units individually), and more consideration heads (12 and 16 separately) than the transformer engineering recommended in the reference [1]. It contains 512 secret units and 8 consideration heads.

The code for BERT:

```
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased', do_lower_case=True)
encoded_data_train =
tokenizer.batch_encode_plus(df[df.data_type=='train'].Keyword.values.astype(str),
add_special_tokens=True, return_attention_mask=True, pad_to_max_length=True,
max_length=256, return_tensors='pt')
input_ids_train = encoded_data_train['input_ids']
attention_masks_train = encoded_data_train['attention_mask']
labels_train = torch.tensor(df[df.data_type=='train'].label.values)
```

3. RESULTS AND DISCUSSION

3.1. LSTM model

The paper will discuss the various outputs the model had produced. Confusion matrix is a measure of the performance of a machine learning (ML) classifier. It essentially shows what rate of success and failure it thinks it has achieved for the different classes. As shown in Figure 2, after normalizing the dataset because of unequal class distribution I fit the model, and here is the attached output [29]. The results after each epoch are also mentioned.

As can be seen from the results, the accuracy of 89.44% is pretty decent. Let's have a look at some other output gauges. The model thinks it achieved 97%, 89%, 89%, 82% accuracy in predicting the labels for sale, software, Bizon products, consultancy which is fairly high enough, since the dataset was significantly skewed towards these categories and not so much for the others.

True Labels	Cloud	0.71	0.00	0.18	0.00	0.07	0.00	0.04
	Sale	0.03	0.97	0.00	0.00	0.00	0.00	0.00
	Amenity	0.10	0.00	0.60	0.04	0.16	0.04	0.06
	Software	0.03	0.00	0.00	0.89	0.00	0.01	0.06
	Consultancy	0.03	0.00	0.02	0.03	0.82	0.01	0.09
	Quickfix	0.02	0.00	0.02	0.10	0.02	0.68	0.16
	Products	0.05	0.00	0.01	0.03	0.01	0.01	0.89
		Cloud	Sale	Amenity	Software	Consultancy	Quickfix	Product
		Predicted Labels						

Figure 2. Confusion matrix

```
epochs = 5
batch_size = 128
history = model.fit(x_sm, y_sm, epochs=epochs,
                    batch_size=batch_size, validation_split=0.2,
                    shuffle = True,
                    callbacks=[EarlyStopping(monitor='val_loss', patience=3,
min_delta=0.00001)])
```



```
Epoch 1/5
274/274 [=====] - 76s 268ms/step - loss: 1.2632 - accuracy: 0.5552
- val_loss: 0.6334 - val_accuracy: 0.7519
Epoch 2/5
274/274 [=====] - 74s 268ms/step - loss: 0.4294 - accuracy: 0.8569
- val_loss: 0.4441 - val_accuracy: 0.8448
Epoch 3/5
274/274 [=====] - 74s 270ms/step - loss: 0.3014 - accuracy: 0.8991
- val_loss: 0.3014 - val_accuracy: 0.8981
Epoch 4/5
274/274 [=====] - 74s 270ms/step - loss: 0.2409 - accuracy: 0.9207
- val_loss: 0.3479 - val_accuracy: 0.8888
Epoch 5/5
274/274 [=====] - 74s 271ms/step - loss: 0.2137 - accuracy: 0.9291
- val_loss: 0.3570 - val_accuracy: 0.8956
```

From this, a few other important performance measures can be derived. Like precision, recall, F1 score, and support. The results of the aforementioned measures are given as:

Classification report:

	precision	recall	f1-score	support
Cloud	0.24	0.71	0.36	28
Sale	1.00	0.97	0.99	73
Amenity	0.80	0.60	0.69	100
Software	0.90	0.89	0.90	405
Solutions	0.76	0.82	0.79	116
Quickfix	0.81	0.68	0.74	94
Bizon Products	0.92	0.89	0.90	681
accuracy			0.85	1497
macro avg	0.78	0.80	0.77	1497
weighted avg	0.88	0.85	0.86	1497

3.2. The transformer: BERT

An epoch in ML means one complete pass of the training dataset through the algorithm. This epochs number is an important hyperparameter for the algorithm. It specifies the number of epochs or complete passes of the entire training dataset passing through the training or learning process of the algorithm. The epoch outputs of the BERT are shown as:

```
Epoch 1: 100%
4241/4241 [24:04<00:00, 3.34it/s, training_loss=0.000]
Epoch 1
Training loss: 0.7308946166182062
Validation loss: 0.5069296853073614
F1 Score (Weighted): 0.864866316429029

Epoch 2: 100%
4241/4241 [23:58<00:00, 3.40it/s, training_loss=0.003]
Epoch 2
Training loss: 0.4657201892458879
Validation loss: 0.5227914065128355
F1 Score (Weighted): 0.8865422382243129

Epoch 3: 100%
4241/4241 [23:55<00:00, 3.40it/s, training_loss=0.001]
Epoch 3
Training loss: 0.34112237018749125
Validation loss: 0.570617884422414
F1 Score (Weighted): 0.8881344253714939
```

Accuracy per class on validation:

Class: Bizon Products Accuracy: 956/1029	Class: Cloud Accuracy: 34/39	Class: Amenity Accuracy: 105/135	Class: Sale Accuracy: 88/88
Class: Consultancy Accuracy: 139/167	Class: Software Accuracy: 576/637	Class: Quickfix Accuracy: 115/150	

3.3. Output model

As shown in Table 1, the outputs are reproduced in not only a first higher level of categorization but also in terms of brand. For the cases of subcategories and brands, it's not possible to train neural networks to predict 1 label correctly out of a possible 300. There are lots of subcategories and brands for every product Bizon offers. So, in order to solve this, the research method has used the TF-IDF vectorizer approach to essentially identify and generate similarity between queries and training datasets on the basis of sequences of letters. So, for order codes and subcategories, this approach would yield maximum efficiency. Neural networks simply won't work because you have to add as many dense layers [30]. Let's assume you assume 300 dense layers. The complexity of the model would be ginormous and the accuracy would fall flat.

Table 1. Output model

User queries	Predicted label	Subcategory	Brand	Order code
webcam a980ji1	Software	web cameras		a980ji1
Macbook 2022	Bizon products	portable computing systems	Apple	
deals on headphone	Sale	Headphone deals		
AWS ec-2	Cloud	Ec-2 systems		
Black friday deals	Sale	Sale		
laptop	Bizon products	laptop computers	lenovo	
driver issues	Quickfix	Driver support		
Asus ROG h19k0nm	Bizon products	laptop computers	xps	h19k0nm
Adobe Premiere	Software	Software Licenses	Adobe	
nintendo switch	Software	keyboard and mice combos		
Windows 11 License	Software	Operating system	Windows	

4. CONCLUSION

The main contribution of this paper is to simplify aspects of analyzing the longtail of keywords that are ignored when it comes to text analytics. Multi-class classification is truly an incredulous tool to capture the sentiments of user queries. When one considers MNC's one has to have a look at how the business plan reacts to customer demands. The better the business understands the demands of customers, the more profit they can reap and that's the main objective is to become profitable and sustainable. The future scope of the work is to produce it and to get an accuracy of >95%. BERT would play a crucial role in getting the accuracy

up since the technology it uses is bleeding edge. Further, the use of back-propagation to self-correct incorrectly classified categories is also on the agenda to be implemented.

ACKNOWLEDGEMENTS

This research work was funded by “Woosong University’s (Daejeon, Republic of Korea) Academic Research Funding-2023”.





REFERENCES

- [1] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *Journal of Big Data*, vol. 2, no. 1, pp. 1–22, Dec. 2015, doi: 10.1186/s40537-014-0007-7.
- [2] A. Banerjee, T. Bandyopadhyay, and P. Acharya, “Data analytics: hyped up aspirations or true potential?,” *Vikalpa*, vol. 38, no. 4, pp. 1–12, 2013, doi: 10.1177/0256090920130401.
- [3] P. Yuan, Y. Chen, H. Jin, and L. Huang, “MSVM-kNN: combining SVM and k-NN for multi-class text classification,” in *IEEE International Workshop on Semantic Computing and Systems*, Jul. 2008, pp. 133–140, doi: 10.1109/WSCS.2008.36.
- [4] A.-S. Dadzie and E. Pietriga, “Visualisation of linked data – reprise,” *Semantic Web*, vol. 8, no. 1, pp. 1–21, Nov. 2017, doi: 10.3233/SW-160249.
- [5] J. Chen, H. Huang, S. Tian, and Y. Qu, “Feature selection for text classification with Naïve Bayes,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 5432–5435, Apr. 2009, doi: 10.1016/j.eswa.2008.06.054.
- [6] M. D. N. Arusada, N. A. S. Putri, and A. Alamsyah, “Training data optimization strategy for multiclass text classification,” in *2017 5th International Conference on Information and Communication Technology (ICoICT)*, May 2017, pp. 1–5, doi: 10.1109/ICoICT.2017.8074652.
- [7] B. Li and C. Vogel, “Improving multiclass text classification with error-correcting output coding and sub-class partitions,” in *Canadian Conference on Artificial Intelligence*, 2010, pp. 4–15, doi: 10.1007/978-3-642-13059-5_4.
- [8] Q. Li and L. Chen, “Study on multi-class text classification based on improved SVM,” in *Advances in Intelligent Systems and Computing*, vol. 279, Springer, 2014, pp. 519–526.
- [9] C. Du, Z. Chen, F. Feng, L. Zhu, T. Gan, and L. Nie, “Explicit interaction model towards text classification,” in *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, 2019, pp. 6359–6366, doi: 10.1609/aaai.v33i01.33016359.
- [10] Z. Li, W. Shang, and M. Yan, “News text classification model based on topic model,” in *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, 2016, pp. 1–5.
- [11] R. Bose, “Understanding management data systems for enterprise performance management,” *Industrial Management & Data Systems*, vol. 106, no. 1, pp. 43–59, 2006.
- [12] D. Pal, M. Mitra, and S. Bhattacharya, “Exploring query categorisation for query expansion: a study,” *arXiv preprint arXiv:1509.05567*, vol. 1, pp. 1–20, Oct. 2015, doi: 10.48550/arXiv.1509.05567.
- [13] G. Andrienko et al., “Space, time and visual analytics,” *International Journal of Geographical Information Science*, vol. 24, no. 10, pp. 1577–1600, 2010, doi: 10.1080/13658816.2010.508043.
- [14] S. Majaro, *International marketing (RLE international business): a strategic approach to world markets*. Routledge, 2013.
- [15] Y. Shi, K. Yao, L. Tian, and D. Jiang, “Deep LSTM based feature mapping for query classification,” in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1501–1511, doi: 10.18653/v1/N16-1176.
- [16] K. Sreelakshmi, P. C. Rafeeqe, S. Sreetha, and E. S. Gayathri, “Deep bi-directional LSTM network for query intent detection,” *Procedia Computer Science*, vol. 143, pp. 939–946, 2018, doi: 10.1016/j.procs.2018.10.341.
- [17] N. T. Thomas, “A LSTM based tool for consumer complaint classification,” in *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018*, 2018, pp. 2349–2351, doi: 10.1109/ICACCI.2018.8554857.
- [18] S. González-Carvajal and E. C. Garrido-Merchán, “Comparing BERT against traditional machine learning text classification,” *arXiv preprint arXiv:2005.13012*, 2020, doi: 10.48550/arXiv.2005.13012.
- [19] P. A. Boda and B. Brown, “Priming urban learners’ attitudes toward the relevancy of science: a mixed-methods study testing the importance of context,” *Journal of Research in Science Teaching*, vol. 57, no. 4, pp. 567–596, Apr. 2020, doi: 10.1002/tea.21604.
- [20] C. C. Aggarwal and C. Zhai, “A survey of text classification algorithms,” in *Mining Text Data*, Springer, 2012, pp. 163–222, doi: 10.1007/978-1-4614-3223-4_6.
- [21] M. Wadnerkar and D. R. Ingle, “Supporting privacy protection in personalized web searching and browsing,” *International Journal of Computer Science and Information Technologies (IJCSIT)*, vol. 6, no. 4, pp. 4086–4093, 2015.
- [22] A. Sadeghi, G. Wang, and G. B. Giannakis, “Deep reinforcement learning for adaptive caching in hierarchical content delivery networks,” *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1024–1033, 2019, doi: 10.1109/TCCN.2019.2936193.
- [23] J. Weston, S. Chopra, and A. Bordes, “Memory networks,” *arXiv preprint arXiv:1410.3916*, 2015, doi: 10.48550/arXiv.1410.3916.
- [24] P. Poonia and V. K. Jain, “Short-term traffic flow prediction: using LSTM,” in *2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3)*, Feb. 2020, pp. 1–4, doi: 10.1109/ICONC345789.2020.9117329.
- [25] E. Çano and M. Morisio, “Word embeddings for sentiment analysis: a comprehensive empirical survey,” *arXiv preprint arXiv:1902.00753*, Feb. 2019.
- [26] M. Xiaofeng, W. Wei, and X. Aiping, “Incorporating token-level dictionary feature into neural model for named entity recognition,” *Neurocomputing*, vol. 375, pp. 43–50, 2020, doi: 10.1016/j.neucom.2019.09.005.
- [27] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, pp. 4171–4186, 2019, doi: 10.18653/v1/N19-1423.
- [28] Tanwar, S., Nayyar, A., & Rameshwar, R. (Eds.). (2021). *Machine Learning in Signal Processing: Applications, Challenges, and*





- the Road Ahead (1st ed.). Chapman and Hall/CRC. <https://doi.org/10.1201/9781003107026>
- [29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Commun. ACM* 60, 6 (June 2017), 84–90. <https://doi.org/10.1145/3065386>
- [30] S. Meng and W. T. Chu, “A study of garbage classification with convolutional neural networks,” in *Indo - Taiwan 2nd International Conference on Computing, Analytics and Networks, Indo-Taiwan ICAN 2020 - Proceedings*, 2020, pp. 152–157, doi: 10.1109/Indo-TaiwanICAN48429.2020.9181311.

BIOGRAPHIES OF AUTHORS







Padarabinda Palai     He received his B.Tech degree in computer science engineering from International Institute of Information Technology, Bhubaneswar in 2021. He is adept in language processing as well as data processing techniques. His current research interests include optimization algorithms and cutting-edge language processing methods. He is currently serving as a software engineer in a Fortune top 50 company. He can be contacted at email: b217026@iiit-bh.ac.in.







Kaushiki Agrawal     Received B.Tech in Computer Engineering degree from International Institute of Information Technology, Bhubaneswar in 2020. She is currently working in a top multinational company from Fortune 50 as a software developer. Her currently aspires to do masters in computer science with her research interest in web 3.0, machine learning, cloud. She can be contacted at email: b516024@iiit-bh.ac.in.



Debani Prasad Mishra     received the B.Tech in electrical engineering from the Biju Patnaik University of Technology, Odisha, India, in 2006 and the M.Tech in power systems from IIT, Delhi, India in 2010. He has been awarded the Ph.D. degree in power systems from Veer Surendra Sai University of Technology, Odisha, India, in 2019. He is currently serving as Assistant Professor in the Department of Electrical Engineering, International Institute of Information Technology Bhubaneswar, Odisha. His research interests include soft computing techniques application in power system, signal processing and power quality. He can be contacted at email: debani@iiit-bh.ac.in.



Surender Reddy Salkuti     received the Ph.D degree in Electrical Engineering from the Indian Institute of Technology, New Delhi, India, in 2013. He was a Postdoctoral Researcher with Howard University, Washington, DC, United State Amerika, from 2013 to 2014. He is currently an Associate Professor with the Department of Railroad and Electrical Engineering, Woosong University, Daejeon, South Korea. His current research interests include market clearing, including renewable energy sources, demand response, smart grid development with integration of wind and solar photovoltaic energy sources. He can be contacted at email: surender@wsu.ac.kr.