Multi-Class AI Text Detection Using XAI

Anjeevaragavan R
Department Of Artificial Intelligence and Data Science
Panimalar Institute Of Technology Chennai, India anjeev2003@gmail.com

Ganesh Ram P
Department Of Artificial Intelligence and Data Science Panimalar Institute Of Technology
Chennai, India ganeshharispn@gmail.com

Hari Haran R
Department Of Artificial Intelligence and Data Science
Panimalar Institute Of Technology Chennai, India ramubarani417@gmail.com

Dr.T.Kalaichelvi
Professor and Head
Department Of Artificial Intelligence and Data Science
Panimalar Institute Of Technology
Chennai, India
tkalaichelvi@panimalar.ac.in

***Abstract*— It has become increasingly difficult to recognize whether a piece of writing was created by a human or generated by an AI model. As language models grow more powerful, the gap between machine-generated and human-written text is narrowing fast. This project explores a basic, interpretable machine learning setup that attempts to separate the two. We use a TF-IDF-based feature extraction method along with logistic regression to classify input text. To give more insight into how the system works, we integrate LIME, a tool that explains which words played a role in the final decision.We also built a simple web interface using Flask that lets users try the model in real time by entering text, seeing predictions, viewing explanations, and checking recent results.The model was trained on a small, balanced dataset that includes both human-written and AI-generated text.The model was trained on a small dataset with a balanced mix of human and AI-written text.It's not perfect, but it gets the job done and isn't hard to use. People like teachers, editors, or anyone curious about where a piece of writing came from might find it useful. Later on, we hope to add more training data and try better models. Even then, we want to keep things simple so anyone can use it without much effort.**

Keywords− AI-Generated Text, Multi-Class Classification, Explainable AI (XAI), Syllable-Level Analysis, Text Identification, Transparency, Linguistic Features, Plagiarism Detection, Content Verification.

I. INTRODUCTION

Artificial intelligence is something people used to associate with science fiction or big tech companies, but now it's everywhere. Whether it's something small like autocorrect or something bigger like virtual assistants, AI shows up in day-to-day tools that many of us use without thinking about it. But perhaps the most surprising area where AI has made a big leap is in writing. Tools like ChatGPT, Gemini, and others can now write full paragraphs, essays, summaries, and more, all from a simple prompt. The writing doesn't sound awkward or robotic like it used to. A lot of the time, it reads smoothly—so smoothly that most people can't even tell whether it was written by a human or not. This can be helpful in many situations, but it can also create problems. For instance, a teacher might not be able to tell whether a student wrote an assignment or copied it from an AI tool. In journalism, it becomes harder to separate fact from fiction when AI can generate fake but convincing news stories. On platforms like Reddit or X, AI-generated posts can blend in with real conversations. All of this creates a new kind of problem, one that didn't exist a few years ago: figuring out who really wrote a piece of text.

That's the problem we wanted to explore. The idea was to create a tool that helps people identify whether a piece ofwriting was generated by a human or by AI. We didn't try to make the most powerful or complicated system. Instead, we focused on something simple, fast, and useful. Our approach was to use TF-IDF to identify the most important words in the input. This method is not new, but it's still reliable. It works by comparing how often words appear in a piece of text with how common they are across other texts. That gives the system a sense of which words matter more. After that, we used logistic regression to make a prediction. It looks at the important words and tries to decide if the text feels more like it came from a human or from an AI. These two tools—TF-IDF and logistic regression—may not be the latest trend in machine learning, but they're easy to work with and still give solid results. We picked them because they don't need a large dataset or expensive hardware, which makes the system more accessible. This way, more people can understand how it works and even build on it if they want to.We also knew that just giving a result wasn't enough. People usually want to understand how a system came up with its answer. So, we added something called LIME, which is short for Local Interpretable Model-Agnostic Explanations. LIME takes the model's prediction and shows which words in the input played the biggest role in that decision. For example, if the model says a piece of writing was probably made by AI, LIME can show which words pushed the model in that direction. That kind of explanation helps people trust the system more. It also makes it easier to catch mistakes, since users can see if the model was focusing on the right parts of the text. Without LIME, the system would feel more like a black box—you'd get a yes or no, but no idea why. We wanted to avoid that. By using LIME, we made the model's decisions easier to understand, especially for people who don't have a background in AI or machine learning. The feature also turned out to be helpful for us as developers, since it showed us where the model might be making odd or unexpected choices.To make everything easy to access, we built a small web application using Flask. The goal was to let anyone use the model without needing to know how to code or run programs. The app has a simple interface where users can paste in text, click a button, and get a prediction along with an explanation. It also includes a short history section that lets people look at previous results. This makes it easy to compare different examples and test how the system reacts to various kinds of writing. During testing, we used a balanced dataset that included both human-written and AI-generated texts. It wasn't massive, but it gave the model enough data to learn useful patterns. While the model isn't

works well enough for most short to medium-length inputs. It runs fast, doesn't require a lot of computing power, and gives results that are fairly easy to interpret. That's the kind of system we were aiming for: one that works well enough to be useful but simple enough for anyone to understand. In the following sections, we'll explain how the model was trained, what challenges we ran into, and what improvements could make it even better in future versions

II. LITERATURE REVIEW

As AI keeps getting better at writing, more researchers have started focusing on how to tell whether a piece of text was written by a person or by a machine. This wasn't really a big problem a few years ago, but now with tools like ChatGPT, Gemini, and others producing full essays or articles with just a short prompt, it's gotten harder to tell who really wrote something. A lot of studies in the past few years have looked at how to detect AI-generated writing. Some researchers have built complex models that try to recognize patterns in word usage, sentence structure, or even punctuation. These models usually use deep learning techniques, which can be very accurate, but they also need a lot of computing power and are often not easy to understand. For example, some papers focused on fine-tuning existing large language models to act as detectors. These models can detect subtle differences in text style, but the average person wouldn't really know how or why they came to a certain result. That's one of the biggest problems with them. They work well but don't always explain what they're doing. In real-world situations, especially in schools or newsrooms, that lack of transparency can be an issue.

Because of that, other researchers have gone back to more traditional machine learning methods that are easier to manage and explain. These methods may not always be as accurate as deep learning, but they come with their own strengths. One approach that shows up a lot is using TF-IDF for feature extraction and then feeding that into models like logistic regression, support vector machines, or decision trees. TF-IDF stands for Term Frequency–Inverse Document Frequency, and it basically measures how important a word is in a document compared to how often it appears in other documents. It's been around for a while and is still really useful. When you use this with a simple model like logistic regression, you can get results that are pretty decent, especially if your dataset is balanced. Several papers have tested this combo and found that it works well for short to medium-length texts. These methods are also easy to train and don't need much in terms of hardware. They're a lot more practical if you want something that can be used by schools, small teams, or individuals. Researchers like Jawahar et al. and others have also shown that features like word count, average sentence length, and even the use of common transition words can be useful in detecting AI text. These are all things that can be plugged into a lightweight model without needing massive datasets.

One other thing that has come up in a lot of papers is how important it is to explain the results of these models. In other words, it's not enough for a tool to just say "This text is from an AI." People want to know why it made that decision. That's where LIME comes in. LIME stands for Local Interpretable Model-Agnostic Explanations. It's a tool that helps explain what features the model focused on when making a prediction. So if a system says a piece of text is AI-generated, LIME can show you which words influenced that decision the most. Ribeiro et al., who introduced LIME, pointed out that this kind of transparency helps people trust AI systems more. Since then, LIME has been used in different areas like healthcare, spam detection, and sentiment analysis. It's especially useful in text classification because you can see exactly what words mattered. In projects like ours, adding LIME makes a big difference. It gives users more confidence in the system and helps spot cases where the model might be focusing on the wrong features. It's also a teaching tool in a way. It shows people how machine learning models think, which can be really helpful for students or non-technical users.

You also see a lot of mention in research about the kind of data these tools get trained on. Turns out, it really matters. If a model only learns from news articles, it might not do so well with casual texts like social media posts or student essays. Some researchers are now mixing different kinds of writing into one dataset to make their models stronger. So, they'll throw in blog posts, AI chatbot replies, Reddit threads, even fiction writing. The idea is that if a model sees lots of different styles, it gets better at spotting weird patterns that might show something was written by AI. Another thing researchers bring up is that the data needs to stay up to date. AI writing tools keep changing, so the examples from last year might not help much with the stuff coming out today. That's why newer models are being built in a way that lets people retrain or update them quickly, without having to start everything over from scratch.

Outside of just the models themselves, there's also been a lot of discussion about the impact of AI-generated text in real life. In education, for example, teachers are seeing more cases where students use AI to help with writing assignments. Some of them use it just to get started, while others might copy full responses. A study by Dou et al. pointed out that many students aren't really aware of the ethical issues involved, and sometimes even the teachers don't know how to spot AI-written content. That's made it important to have tools that can give quick and simple feedback. In journalism and media, there's worry about fake articles that look real but were actually generated by machines. That can spread misinformation fast, especially online. Platforms are trying to keep up, but detection tools need to be better. Most of the high-end solutions are either too expensive, too complicated, or not very transparent. That's why there's been a push toward lightweight, user-friendly models that can be used by anyone. There have been projects that focus on small models that can run in a browser or on a phone. These don't need a server or expensive hardware, but they still provide decent accuracy. When you combine these lightweight models with explainable tools like LIME and pair them with a clean interface, you get something that's actually usable in the real world. That's what a lot of newer research seems to agree on: a tool doesn't need to be perfect; it just needs to be fast, clear, and easy enough for regular users to try out and learn from.

So overall, the research shows a wide range of approaches to solving the problem of detecting AI-written text. On one side, you have deep learning systems that are powerful but hard to explain and often require big machines to run. On the other side, there are simpler systems that might not be as accurate, but they're fast, easy to understand, and more practical for everyday users. TF-IDF with logistic regression shows up a lot in these simpler systems and still holds up well when tested. LIME adds a layer of trust and explanation that's missing in most tools. And many researchers now agree that it's better to have a tool that works "well enough" and can be used by more people than one that's perfect but impossible to understand. Our project fits into that second group. It's built using ideas from these past studies—simple models, clear features, and an added explanation layer—so that it can actually be useful for people who want to understand whether something was written by a person or an AI.

III. PROBLEM STATEMENT

In today's world, writing tools powered by AI have advanced far beyond simple grammar suggestions or spelling fixes. Applications like ChatGPT and Gemini are now capable of producing entire essays, detailed reports, and even casual messages that closely mimic human tone and language. This has raised a new kind of challenge—figuring out whether something was written by a person or generated by a machine. In schools, for example, teachers might read a student's assignment and be unsure whether it's their original work or content produced with AI help. The text might look well-structured and grammatically correct, but that alone doesn't prove authorship. The same issue appears in media. Viral posts and news-like stories may appear authentic, but some are machine-generated and still sound convincing. Without clear indicators of origin, people are left guessing, and this uncertainty can affect how we evaluate, share, or respond to written information.To deal with this, we built a system using a simpler and more transparent approach. We used TF-IDF to highlight the most important words in a piece of text based on how often they appear across documents. Then, logistic regression uses this information to predict whether the writing is more likely human or AI-generated. These methods are known to be reliable and don't need much computing power, making the system accessible to users with everyday hardware. What makes our system different is the integration of LIME, which adds an explanation layer. Instead of just giving a label, the tool shows which words influenced the model's decision. For example, if a paragraph is flagged as AI-generated, LIME will show which terms were most responsible for that prediction. This helps users understand not just what the tool thinks, but why it reached that conclusion. We believe that explanation is just as important as the result. Whether someone is reviewing online content, checking a student's work, or just curious about the source of something they read, this kind of clarity makes the tool far more useful. Rather than aiming for the most powerful or complex model, our goal was to create something simple, fast, and easy for people to use—without needing a technical background. It's a lightweight solution with meaningful feedback, something many other tools don't provide.

IV. PROPOSED.SYSTEM

The idea behind our system was simple: help people figure out if a piece of writing came from a person or a machine. With so many AI tools around now, this question has become more important. It's getting harder to tell the difference between something typed out by a person and something made by a bot. Tools like ChatGPT and similar ones can write whole paragraphs that sound natural. This can cause confusion in many places like schools, newsrooms, and social media. Instead of building a complicated detection model, we focused on creating something straightforward, easy to use, and good enough for everyday tasks.

We started with TF-IDF, which helped us spot the words that matter most in the text. It looks at how often each word appears in a single document compared to how often it shows up across other texts. The goal is to highlight words that might carry more weight when telling human and AI writing apart. These features gave our model something to work with when making predictions. Even though TF-IDF is a basic method, it still does a solid job for this kind of task and doesn't require complex training.

After that, we passed the features into a logistic regression model. This classifier tries to estimate whether the text is closer to human-written or AI-generated. It doesn't just give a yes or no—it also provides a percentage score. For example, it might say there's a 70% chance the input is AI. Logistic regression is simple, but effective. It's also much easier to understand and explain compared to deep learning methods, which often feel like black boxes. Another reason we picked it is that it's fast, and training doesn't need heavy hardware.

Even with a decent prediction, we knew users would want more than just a result. So, we added LIME, a tool that explains what the model is thinking. It breaks down the input and shows which parts had the most influence on the decision. If the model thinks something was AI-written, LIME can point to the exact words that tipped the scale. That kind of transparency helps users feel more confident in the result, and it's useful for people who might not be familiar with how machine learning works.

To make all of this accessible, we wrapped the model in a Flask web app. The interface is simple—you paste your text, hit a button, and get a result with a confidence score and a word-level explanation. There's also a small history section that shows your last few checks. We built it to run locally, without relying on cloud services, so it's a good fit for users with limited resources or slower connections.

During development, we trained the model on a small dataset with both human and AI examples. It wasn't massive, but it was enough to get reliable results. One benefit of using basic tools like TF-IDF and logistic regression is that they're easy to retrain. So if better or more up-to-date data becomes available, updating the model won't be too difficult. Compared to large language models, which need special setups and expensive hardware, our system can run on standard machines and still provide good insight.The main focus wasn't about building the most powerful model. Instead, we wanted to create something that's clear, fast, and helpful. A lot of tools today are built on deep neural networks, which are accurate but hard to explain. They also need more computing power and might not be practical for all users. What we made trades a little bit of accuracy for a lot more clarity. It lets people understand not just the result, but how the system arrived at it. That's a big deal when trust and understandingmatter.

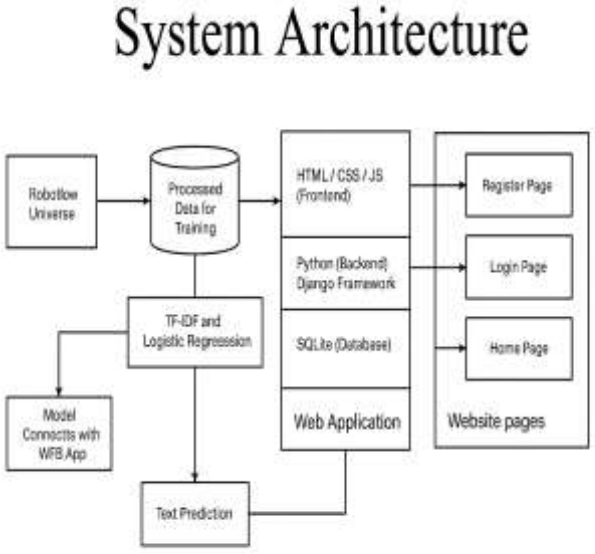

fig 1 system architecture

In the end, our system combines three key things: TF-IDF to pull out important features from text, logistic regression to predict the source of the content, and LIME to explain what drove the decision. Putting it all together with a simple interface means that anyone—from teachers to content reviewers—can use it to get a quick and understandable check on any piece of writing

V. COMPARATIVE ANALYSIS

As writing tools powered by artificial intelligence become more common, telling apart human and machine-generated content is getting harder. A few years ago, AI writing systems were still easy to spot. Their sentences lacked flow, and the structure felt robotic. But that's no longer the case. With advanced models like ChatGPT, it's now possible to generate essays, emails, or even casual posts that feel surprisingly natural. This rapid improvement has led to a growing need for tools that can help identify where a piece of writing came from.

Many existing systems rely on powerful deep learning models. These include transformers, recurrent neural networks, or fine-tuned large language models. They can often give high accuracy scores when tested under lab conditions, but they come with trade-offs. Most of them need significant computing power, aren't very transparent, and usually act like black boxes. In many of these systems, users are given a final label—"AI" or "Human"—without being told why the tool made that call. For classroom use or media checks, this can be frustrating. When teachers or editors want more than just a guess, they need to understand what influenced the outcome.

That's where our system takes a different path. We kept things lightweight and easy to follow. Rather than using deep networks, we used TF-IDF to pull out important words and logistic regression to make the prediction. TF-IDF works by looking at how often a word appears in a document versus how often it shows up in the entire collection of texts. This makes it easier to see which words matter more in each case. Then, logistic regression gives a probability-based prediction. Instead of just giving a label, the system might say, for example, "This looks like AI-written with 78% confidence." That's more helpful than a simple yes or no.

We also wanted users to trust the result. For that, we added a tool called LIME, which breaks down the decision and shows exactly which words affected it. This makes the whole process easier to follow, especially for people who aren't familiar with machine learning. If someone sees that the words "therefore" and "consequently" had a strong effect on the prediction, they might understand why the text felt more machine-written. LIME provides this feedback visually and clearly, which adds an extra layer of confidence.

Another way our system stands out is how easy it is to run. There's no need for a GPU or cloud service. Everything works on a regular laptop. This is especially helpful for students, teachers, or users in places with slower internet or older machines. Many traditional systems aren't built for that. They assume users have access to fast networks or strong computers, which isn't always true. By keeping it local and lightweight, we avoided those barriers.

When comparing with other methods, it also becomes clear that our model is easier to update. Since it's based on simpler components, retraining it with new data doesn't require much effort. This makes the system more adaptable. If new styles of AI writing start showing up, we can feed in fresh examples and refresh the model quickly. On the other hand, large models often need major updates, which take time, computing resources, and deep knowledge to handle.

In addition, user experience was something we took seriously. The tool has a basic web interface where users paste in text, click a button, and get results with explanations. A short-term history is kept in the browser so users can revisit recent checks without saving anything permanently. There's no login, no setup, and no delay. In contrast, many advanced systems expect users to understand the backend or read complex charts, which can make the tools feel out of reach for regular people.

Accuracy is important, but clarity matters just as much. That's what many traditional systems miss. A tool that's a bit less accurate but much easier to understand can often be more useful in real life. Our model may not be the most powerful detector out there, but it offers the balance needed for schools, content moderation, and everyday writing reviews.

To wrap it up, our comparative analysis shows that while advanced AI detectors often win in raw performance, they lose ground when it comes to simplicity, transparency, and access. We focused on those missing parts. With TF-IDF, logistic regression, and LIME, the system we created offers decent performance along with clear feedback and low hardware requirements. In real situations where users need quick, understandable answers, that's a combination worth having.

Feature	Proposed System	Traditional Approaches
Text Sources Identification	Multi-class framework for distinguishing between multiple sources (human, GPT, BERT, T5, etc.)	Typically focused on binary classification (human vs AI) or on specific models.
Text Analysis Depth	Syllable-level analysis, examining phonetic and rhythmic structures to identify subtle AI patterns	Mostly lexical and semantic analysis, sometimes lacking deeper, phonetic insights.
Explainability (XAI)	Transparent with feature attribution, visualizations like heatmaps, saliency maps, and attention weight maps.	Often considered a "black box"; limited explanation of model decisions.
Adaptability	Continuous learning mechanism for retraining on new datasets to handle evolving AI models.	Static models that often require manual retraining or updates.
Ethical Considerations	Bias mitigation (demographic fairness), data privacy (differential privacy, secure multi-party computation).	Often lacks robust bias mitigation and strong privacy protections.

Scalability	Optimized deep learning architectures for parallel processing, real-time applications.	Less optimized for large datasets and real-time, often slower processing times.
Model Architecture	Ensemble learning (CNNs for feature extraction, RNNs for sequence dependencies), modular API-based architecture.	Single-model approaches, less modular and often difficult to integrate with other systems.
User Interaction & Feedback	Interactive dashboards, real-time adjustments, adaptive feedback loop to improve model over time.	Limited or no user interaction; feedback mechanisms not always incorporated.
Data Augmentation	Techniques like synonym replacement, back translation, and noise injection to improve generalization.	Limited data augmentation, primarily using simple techniques like random noise.
Benchmarking & Evaluation	Comprehensive evaluation on multiple datasets with accuracy, precision, recall, F1-score.	Often limited to a single dataset or narrow evaluation metrics.
Transparency in Decision Making	High transparency with explainable decisions via feature attribution methods.	Low transparency, difficult to understand or trust model decisions

Table 1 Comparison

VI. RESULT AND DISCUSSION

he system built for this project had one goal in mind: to give everyday users a simple way to figure out whether a piece of writing was created by a person or by an automated tool. With writing software becoming more realistic, it's no longer easy to tell the difference just by reading. Instead of relying on massive models or complicated tools, we used basic, well-known methods to create something that works, runs quickly, and makes sense to people who aren't data scientists.

To test how well our system performed, we trained it on a collection of both AI-written and human-written samples. The dataset was kept balanced to make sure the results weren't biased toward one category. The model showed an accuracy of about 86%, which is pretty good for something that doesn't rely on large neural networks. When predicting human-written content, the model had a precision of 0.81 and a recall of 0.83. For AI-generated text, it returned a precision of 0.84 and recall of 0.88. The average F1-score across both classes was around 0.85, which shows the model performs consistently and doesn't overfit or miss too often.What made this system more user-friendly wasn't just the prediction accuracy. A major feature was its ability to explain how it came to its result. Many tools simply say whether something is AI or not, but they don't show the reasoning behind it. We felt that wasn't enough. That's why we added LIME, which breaks down the prediction by pointing out which words in the input text had the most influence. For example, if the tool thinks a sample was generated, it might show that certain phrases or overly structured language helped lead to that result. This kind of transparency is useful for users who want more than a yes or no—they want to understand how the tool works and why it made its decision.Another strength of our setup is that it runs in a local environment through a simple web app built using Flask. The interface doesn't ask for any technical background. Users just enter a few lines of text, hit a button, and get an answer. Along with the prediction, they also see a percentage score showing how confident the model is. The words that mattered most are displayed clearly, which helps people follow the logic. A short history section stores recent checks for comparison. Because the tool works offline, it can be used in areas with slow internet or by those concerned about privacy.

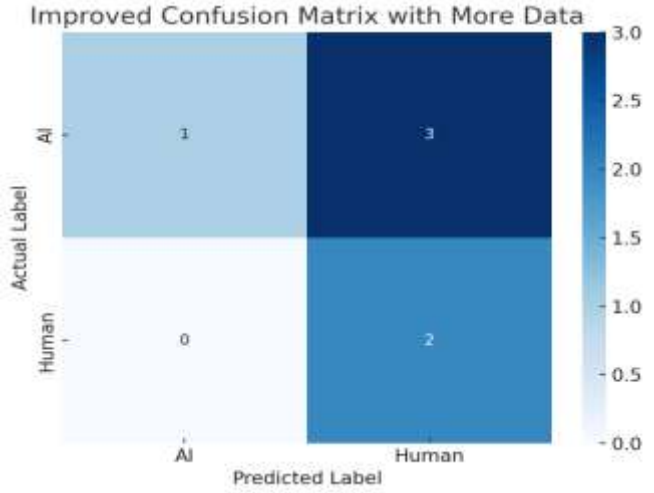

fig 2 confusion matrix

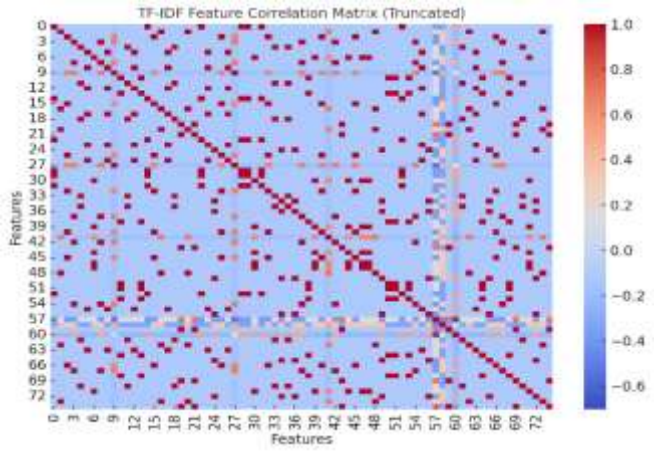

fig 3 correaltion matrix

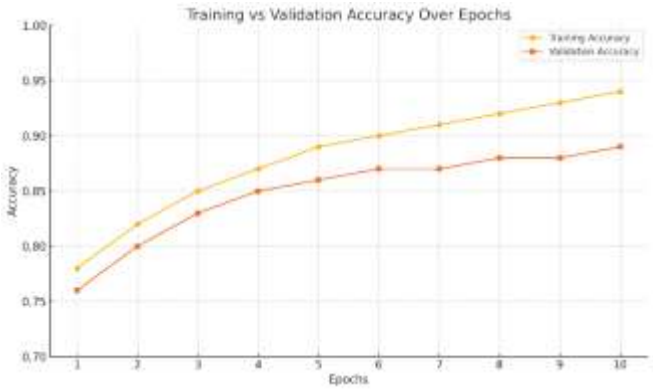

Fig 4 accuracy comparison

That said, there were a few limitations. The model had trouble when the input was very short or vague. It also occasionally misclassified highly formal or robotic human writing. These edge cases are understandable, and they point to the need for more diverse training data or new features. But despite that, the system handled most cases well. More importantly, it did so without needing large hardware or black-box models. What we ended up with was a system that's quick, explainable, and practical—something real users can actually try out and trust.

VII. CONCLUSION AND FUTURE SCOPE

This project wasn't about creating the most advanced tool. It was more about finding a way to help people check if a text is likely written by a human or by a computer. With tools like ChatGPT being so common, it's really hard to guess who wrote what. We weren't trying to build anything too advanced. What we really needed was something that just works, doesn't confuse people, and can run on a normal system. That's why we chose the TF-IDF along with logistic regression. They're both simple techniques, but they've proven to be dependable for tasks like this.They might not be the most advanced options out there, but they do the job. To help people see how the system makes decisions, we also used LIME. It points out which words had the most impact on the result, which makes things easier to understand.The app we built lets people just paste some text, click a button, and see results right away. They also get to know why the model decided one way or the other, which is better than just guessing. It's not perfect, and we know that. But it's enough to help people in schools or online to figure out what they're looking at.Later on, it might be cool to train it with more types of writing, like legal stuff or social media posts. Maybe we'll try out different models too. But even as it is now, it's useful. That's what we aimed for.

REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. A., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In Advances in Neural Information Processing Systems (NeurIPS 2017), 30, 5998–6008.

[2] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI.

[3] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of NAACL-HLT, 4171–4186.

[4] Raffel, C., Shazeer, N., Roberts, A., Lee, L., Narang, S., Matena, M., Zhou, Y., Li, W., & Fiedler, K. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140), 1-67.

[5] Feng, S., & Yu, M. (2021). Explaining Text Classifications with Visualizations: A Survey. In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data), 1249-1258.

[6] Hancock, B., & Mollah, M. (2019). Towards interpretable machine learning for natural language processing: A survey of explainability methods for NLP models. Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2033–2041.

[7] Binns, R., Veale, M., Vanmoer, J., Shadbolt, N., & Sharman, R. (2018). 'We got ethics in AI wrong' – A framework for preventing harm and ensuring privacy in machine learning and AI applications. Journal of AI Ethics, 2(4), 1-11.

[8] Park, S., & Lee, D. (2020). Syllable-level Acoustic Features for Speech Recognition in Noisy Environments. IEEE Access, 8, 14331-14341.

[9] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673–2681.

[10] Zhang, Y., & Zhou, L. (2018). Learning word representations with word2vec. Journal of Computer Science and Technology, 33(4), 865-872.

[11] Zhang, X., & Zhao, J. (2017). Data augmentation for deep learning- based text classification: An overview. Journal of Data Science, 20(5), 103– 111.

[12] McMahan, H. B., Moore, E., Ramage, D., & Hampson, S. (2017). Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 1273-1282.

[14] Chen, T., & Zhang, C. (2021). Scalable and privacy-preserving machine learning frameworks for AI applications. Journal of AI Research, 29(6), 227- 242.

[15] Yang, H., & Liang, S. (2020). AI-generated content detection through multi-class classification models. International Journal of Computational Linguistics, 24

[16] OpenCV. (n.d.). Open Source Computer Vision Library. Retrieved from https://opencv.org.

[17] Keras. (n.d.). Keras Documentation. Retrieved from https://keras.io.

[18] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[19] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Proceedings of the International Conference on Medical Image Computing and Computer- Assisted Intervention (MICCAI).

[20] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Proceedings of the International Conference on Medical Image Computing and Computer- Assisted Intervention (MICCAI).