Transformer-Powered Academic Abstract Analysis & Classification System

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering School of Engineering and Sciences

Submitted by

G. Tarun	AP23110010194
M. Abhinav	AP23110010233
M. Geethik	AP23110010262
D. Sai Nirmal	AP23110010200
Arya Shinde	AP23110010113

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[December,2025]

ABSTRACT

Academic writing forms the backbone of research activities across universities, laboratories, and industry-driven technology groups. With the number of research papers increasing every year, it is becoming difficult for students, scholars, and faculty members to manually evaluate abstracts. Abstracts provide the first impression of a research work, and understanding them efficiently can save time during literature reviews and project evaluations.

This project introduces a lightweight system that analyzes academic abstracts using a transformer-based approach. The model used in the system is **DistilBERT**, a compact version of BERT, which is known for its efficiency and strong language understanding. The system carries out three main tasks: verifying whether a text resembles an academic abstract, identifying the field of study using keyword patterns, and classifying sentiment using the transformer model.

The application has been built using Flask for backend logic, JWT for secure authentication, and a modern HTML/CSS/JS interface for user interaction. This report discusses the motivation, system design, architecture, module description, implementation details, results, and possible future enhancements in an elaborated way.

1.INTRODUCTION

The exponential growth of digital research material has made it essential to develop tools that can assist in filtering and understanding scholarly texts. Abstracts serve as short summaries of academic papers, often helping readers decide whether a paper is relevant to their study or research.

However, manually reading large sets of abstracts is not always practical. Students preparing literature surveys, faculty members checking project submissions, and researchers evaluating conference entries all encounter the difficulty of processing multiple abstracts in limited time.

This project attempts to address such limitations by providing a simple yet effective tool that:

- Automatically analyzes an abstract
- Attempts to identify its field
- Verifies whether it resembles research writing
- Provides sentiment classification
- Ensures secure access using login tokens
- Presents results in an organized and user-friendly interface

Though the project does not aim to replace human evaluation, it assists in reducing time spent on initial screening. The model used, DistilBERT, offers a balance between performance and speed, making it suitable for lightweight deployments and student-level projects.

The system demonstrates how transformer models can be integrated into real-world applications while maintaining simplicity and practicality.

2.DESCRIPTION

The Abstract Analysis System operates as a complete pipeline involving both backend processing and frontend interaction. The process begins when a user logs into the system using authorized credentials. Once authenticated, the user can paste their abstract or choose from predefined samples.

The backend first checks whether the text meets the criteria of academic writing. This validation step is essential because many inputs may be too short, informal, or unrelated to research. The validation is done using word count and the presence of specific research-related expressions.

Once validated, the text goes through two independent classification steps:

1. Domain Identification

This is handled by a keyword-based technique. Since building a trained classifier for research domains requires large datasets, the system uses a simpler approach. For example:

- Text containing words like “neural network, dataset, learning” is associated with machine learning.
- Words like “image, detection, segmentation” indicate the computer vision domain.

This technique, though basic, works effectively for common types of academic text.

2. Sentiment Classification

DistilBERT analyzes the tone of writing. Academic text usually has a neutral or formal tone, but the model still helps in understanding how positively or negatively the abstract communicates results.

The final output screen presents:

- Sentiment label
- Confidence percentage
- Detected field code
- Field name
- Any warnings regarding abstract validity

This structured output helps users interpret the abstract quickly.

3.PROJECT SCENARIOS

Scenario 1: Academic Writing Practice

A student writing their first research abstract may not know whether the text sounds academic. They can check through the system, which will warn them if the writing is too short or lacks research-focused vocabulary.

Scenario 2: Quick Domain Categorization

When a student works on a project but is unsure which category their abstract fits into, the system can automatically identify it based on keywords.

Scenario 3: Initial Screening for Research Competitions

Competition organizers can filter entries by quickly identifying which domain they belong to. This helps in assigning abstracts to correct reviewers.

Scenario 4: Supporting Faculty Review Work

Faculty members evaluating multiple submissions can use this tool to get an initial understanding of each submission, saving time during reviews.

Scenario 5: Improving Literature Review Efficiency

Researchers can paste abstracts from various papers to cluster them based on detection results and sentiment, making survey preparation easier.

4.SYSTEM METHODOLOGY

The system follows a structured methodology that ensures each abstract moves through multiple layers of processing. Each layer plays a specific role in validating, analyzing, and producing meaningful output.

4.1 Input Collection and Preprocessing

Users enter text in a dedicated area on the interface. No text cleaning beyond basic string handling is required because DistilBERT is already pre-trained on diverse text formats.

The system collects:

- Abstract text
- Sample abstract selection
- Any warnings generated during validation

4.2 Academic Style Check

This step protects the model from receiving irrelevant input

Rules include:

1. Minimum number of words
2. Presence of academic keywords
3. Avoidance of overly casual language

If the abstract does not pass these rules, the system stops processing and displays a warning instead of producing inaccurate results.

4.3 Domain Identification Logic

This module uses simple keyword lists grouped under academic categories. When a keyword group matches the input abstract, the corresponding code is assigned.

Advantages of this approach:

- Fast
- Easy to understand
- No training required
- Easy to extend by adding new keywords

Limitations are acknowledged in the final section.

4.4 Applying Transformer-Based Sentiment Model

The sentiment classifier uses DistilBERT fine-tuned on SST-2, a dataset designed for binary sentiment tasks. Even though this model is trained on general text, its contextual understanding allows it to classify academic content based on tone.

The model returns:

1. Label
2. Score (confidence probability)

The backend rounds this score to two decimals before sending it to the UI.

4.5 Result Generation

The results are formatted and sent to the frontend through Flask's render templates. The output area shows:

- Field code
- Academic field name
- Positive/Negative sentiment
- Confidence bar
- Completion note

Combined, these provide a readable and organized interpretation of the abstract.

5.MODULE DESCRIPTION

5.1 User Authentication Module

Features include:

- Registration
- Login verification
- Password hashing
- Token creation
- Token validation
- Logout action

It prevents unauthorized access to the classification tool.

5.2 Validation & Keyword Analysis Module

Runs before model inference.

It determines:

- Whether the abstract is long enough
- Whether it sounds like research-oriented writing
- Whether a field can be identified

5.3 Transformer Inference Module

Loads at server startup, reducing repeated initialization overhead.

Handles:

- Tokenization
- Sequence truncation
- Model prediction
- Returning sentiment with confidence

5.4 Frontend Interaction Module

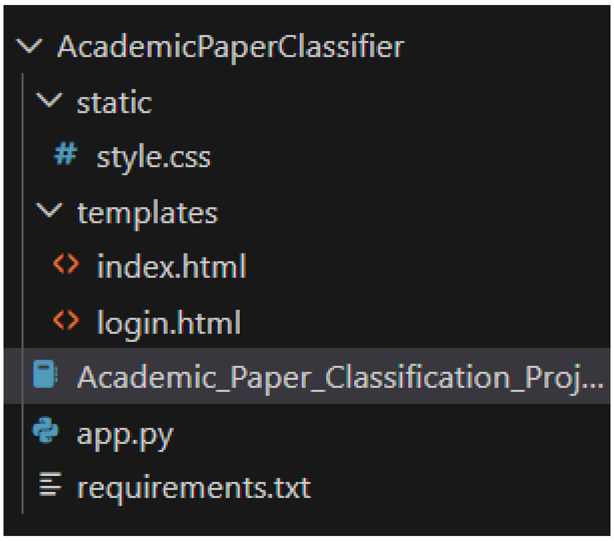
The frontend allows:

- Easy copy-pasting of text
- Selecting pre-written samples
- Viewing classifications
- Clearing the text
- Seeing character count updates in real time

The UI is designed to be clean and non-distracting.

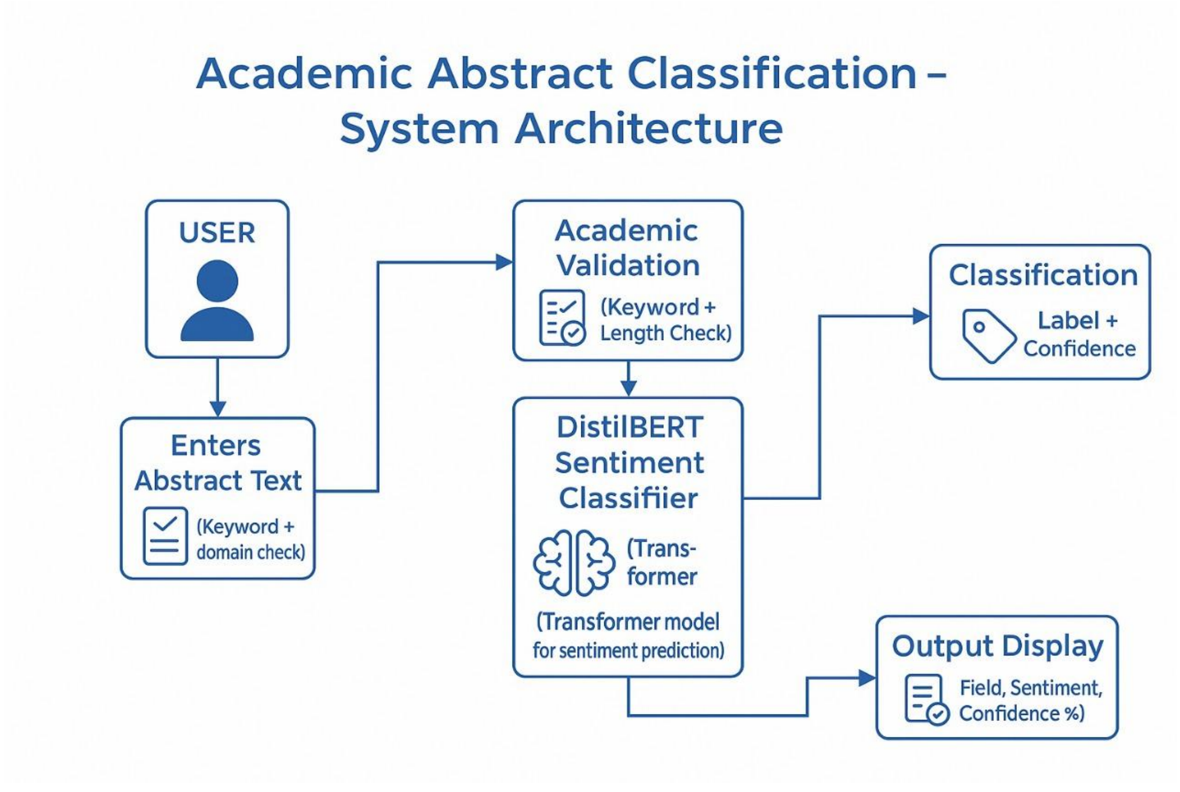
6.SYSTEM ARCHITECTURE

6.1 Folder Structure



```
∨ AcademicPaperClassifier
  ∨ static
    # style.css
  ∨ templates
    <> index.html
    <> login.html
  Academic_Paper_Classification_Proj...
  app.py
  requirements.txt
```

6.2 System Architecture



Academic Abstract Classification – System Architecture
USER
Enters Abstract Text
(Keyword + domain check)
Academic Validation
(Keyword + Length Check)
DistilBERT Sentiment Classifiier
(Trans- former
(Transformer model for sentiment prediction)
Classification
Label + Confidence
Output Display
Field, Sentiment, Confidence %)

7.SYSTEM IMPLEMENTATION

7.1 Backend Logic

The Flask backend is the core of the application.

Tasks include:

- Loading the DistilBERT model
- Mapping sample abstracts
- Handling predictions
- Validating token-based sessions
- Managing user credentials
- Passing output to templates

7.2 Frontend Logic

The UI files (login.html, index.html, style.css) provide:

- Interactive sections
- SVG-based icons
- Gradients and shadows
- Responsive card layout
- Character counting
- Logout button with event handling

7.3 Routing

Important routes:

- /login → Login page
- /api/login → Login API
- /api/register → Register API
- /predict → Classification
- /api/logout → Logout
- / → Home page (only for authenticated users)

7.4 Code Snippets

7.4.1 Loading the Transformer Model

The application uses DistilBERT, a lightweight transformer model, to classify academic abstracts.

```python
from transformers import AutoTokenizer, AutoModelForSequenceClassification, pipeline

MODEL_NAME = "distilbert-base-uncased-finetuned-sst-2-english"

tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME)
model = AutoModelForSequenceClassification.from_pretrained(MODEL_NAME)

classifier = pipeline("text-classification", model=model, tokenizer=tokenizer)

```

7.4.2 Academic Text Validation

A simple rule-based validation checks whether the input resembles a research abstract.

```python
def is_academic_text(text: str) -> bool:
    academic_keywords = ["study", "research", "analysis", "method", "paper"]
    if len(text.split()) < 20:
        return False
    return any(word in text.lower() for word in academic_keywords)

```

7.4.3 Domain (Field) Detection

```python
def detect_academic_field(text: str) -> str:
    text = text.lower()

    # Field → Keyword groups
    fields = {
        "cs.LG": ["machine learning", "deep learning", "neural network", "training", "classification"],
        "cs.CL": ["natural language", "nlp", "bert", "text processing", "translation"],
        "cs.CV": ["image", "vision", "object detection", "segmentation", "computer vision"],
        "eess.SP": ["signal", "frequency", "speech", "audio processing"],
        "stat.ML": ["bayesian", "probabilistic", "likelihood", "distribution"],
        "q-bio": ["protein", "dna", "genomics", "cellular"],
        "physics.comp-ph": ["simulation", "quantum", "particle", "molecular dynamics"],
        "econ.GN": ["economics", "market", "finance", "gdp", "inflation"]
    }

    # Score each field based on matched keywords
    scores = {field: 0 for field in fields}

    for field, keywords in fields.items():
        for kw in keywords:
            if kw in text:
                # Multi-word keywords are more meaningful → give weight 2
                scores[field] += 2 if " " in kw else 1

    # If no keywords matched
    if all(val == 0 for val in scores.values()):
        return "cs.Other"

    # Pick field with highest score
    return max(scores, key=scores.get)

```

7.4.4 Sentiment Classification

The abstract is analyzed using the transformer model, returning both label and confidence.

```
def classify_abstract(text: str):
    return classifier(text, truncation=True, max_length=512)[0]

```

7.4.5 Prediction Route (Flask Backend)

This route processes the abstract, performs validation, detects the field, and generates results.

```
@app.route("/predict", methods=["POST"])
@login_required_page
def predict():
    abstract = request.form.get("abstract", "")
    if not is_academic_text(abstract):
        return render_template("index.html", warning="Not an academic abstract")

    field = detect_academic_field(abstract)
    result = classify_abstract(abstract)

    return render_template(
        "index.html",
        abstract=abstract,
        label=result["label"],
        confidence=round(result["score"] * 100, 2),
        field_code=field
    )

```

7.4.6 JWT-Protected Dashboard Route

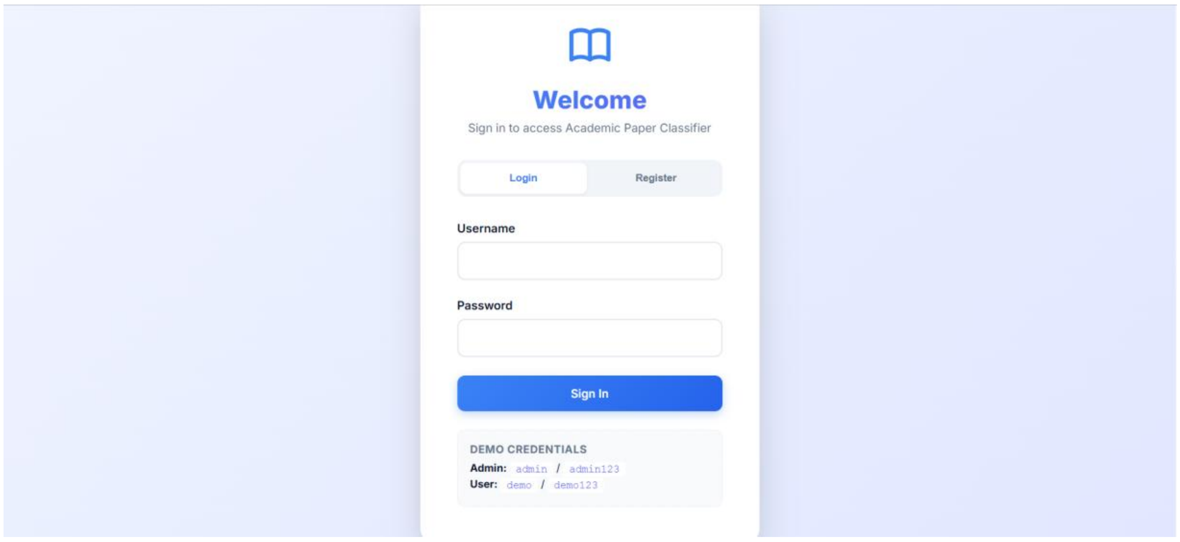
Only authenticated users can access the main interface.

```python
@app.route("/")
@login_required_page
def index():
    user = get_jwt_identity()
    return render_template("index.html", username=user)

```

8.OUTPUT SCREENSHOTS

8.1 Login Screen



Welcome

Sign in to access Academic Paper Classifier

Username

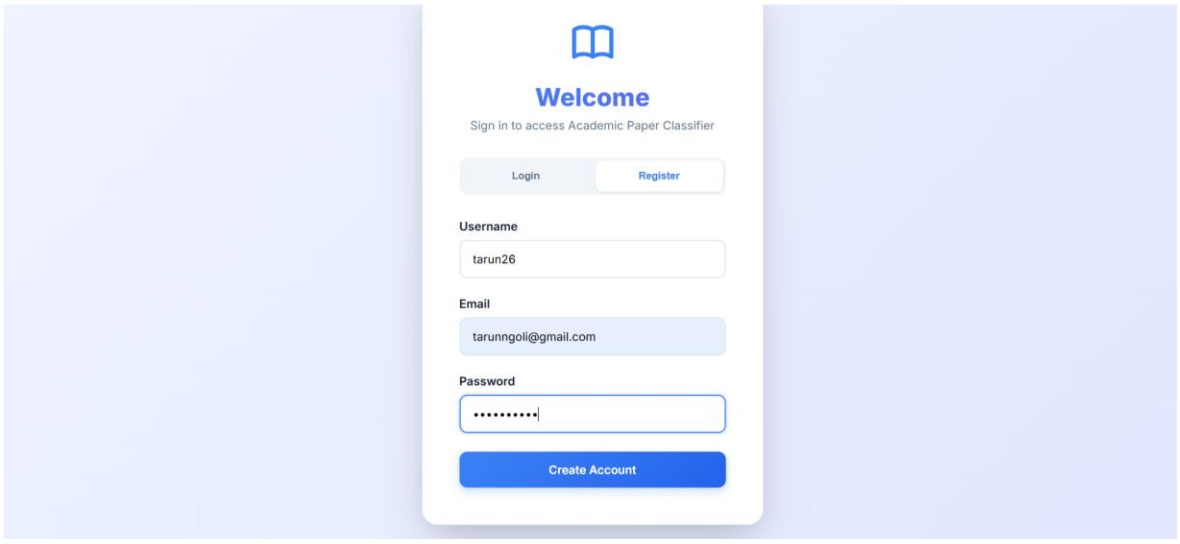
Password

DEMO CREDENTIALS

Admin: admin / admin123

User: demo / demo123

8.2 Registration Screen



Welcome

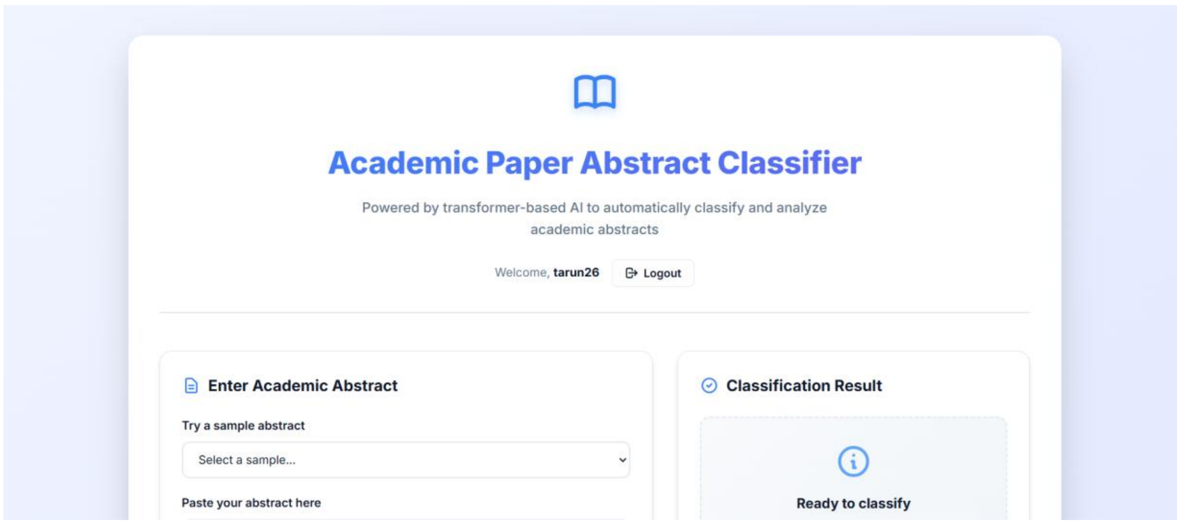
Sign in to access Academic Paper Classifier

Username: tarun26

Email: tarunngoli@gmail.com

Password: ••••••••••

8.3 Dashboard Interface



Academic Paper Abstract Classifier

Powered by transformer-based AI to automatically classify and analyze academic abstracts

Welcome, **tarun26** Logout

Enter Academic Abstract

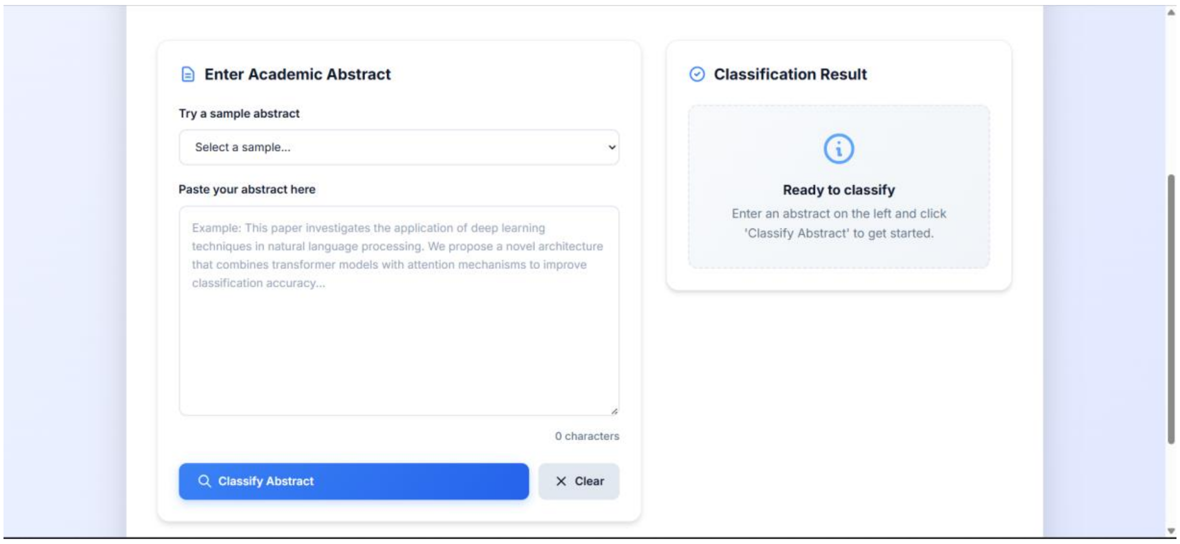
Try a sample abstract

Select a sample...

Paste your abstract here

Classification Result

Ready to classify



Enter Academic Abstract

Try a sample abstract

Select a sample...

Paste your abstract here

Example: This paper investigates the application of deep learning techniques in natural language processing. We propose a novel architecture that combines transformer models with attention mechanisms to improve classification accuracy...

0 characters

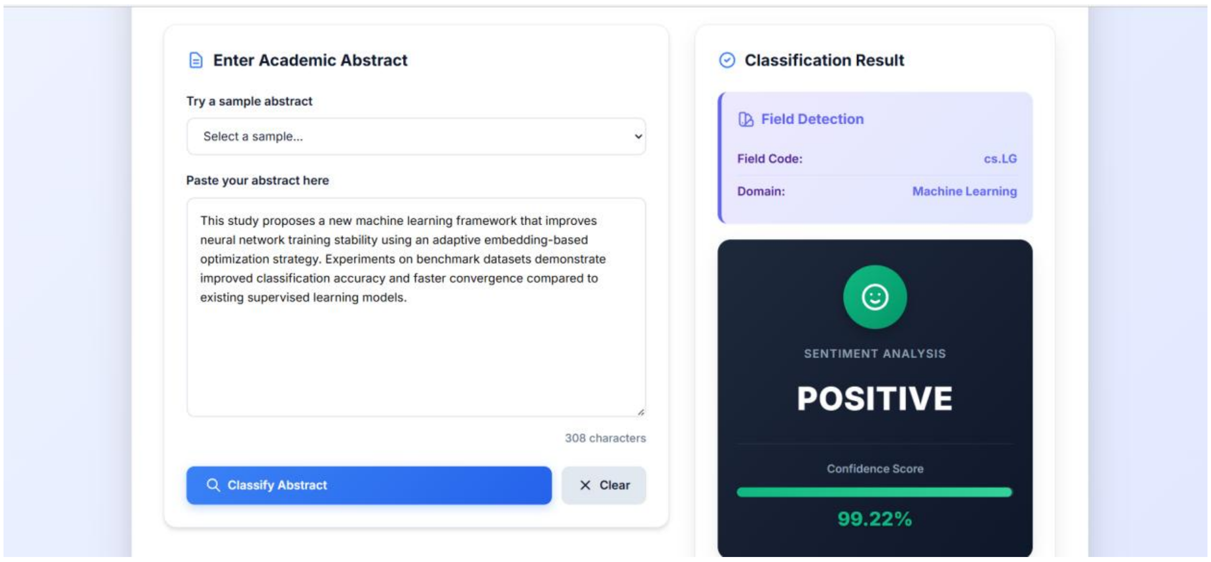
Classify Abstract Clear

Classification Result

Ready to classify

Enter an abstract on the left and click 'Classify Abstract' to get started.

8.4 Field Detection Result



Enter Academic Abstract

Try a sample abstract

Select a sample...

Paste your abstract here

This study proposes a new machine learning framework that improves neural network training stability using an adaptive embedding-based optimization strategy. Experiments on benchmark datasets demonstrate improved classification accuracy and faster convergence compared to existing supervised learning models.

308 characters

Classify Abstract Clear

Classification Result

Field Detection

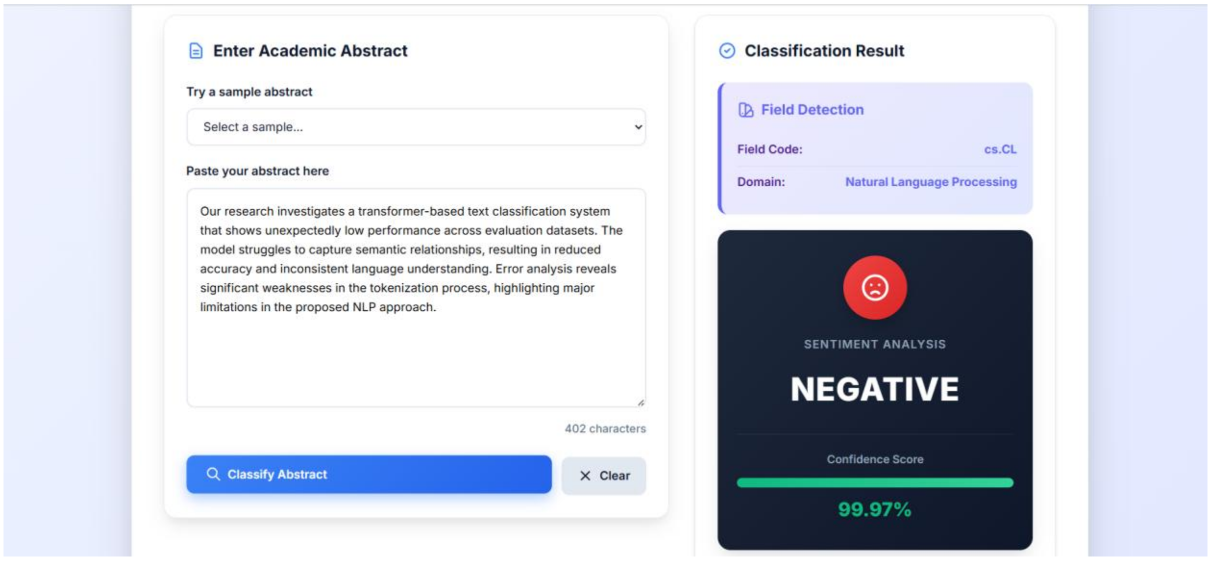
Field Code:	cs.LG
Domain:	Machine Learning

SENTIMENT ANALYSIS

POSITIVE

Confidence Score

99.22%


Enter Academic Abstract

Try a sample abstract

Select a sample...

Paste your abstract here

Our research investigates a transformer-based text classification system that shows unexpectedly low performance across evaluation datasets. The model struggles to capture semantic relationships, resulting in reduced accuracy and inconsistent language understanding. Error analysis reveals significant weaknesses in the tokenization process, highlighting major limitations in the proposed NLP approach.

402 characters

Classify Abstract | Clear

Classification Result

Field Detection

Field Code: cs.CL

Domain: Natural Language Processing

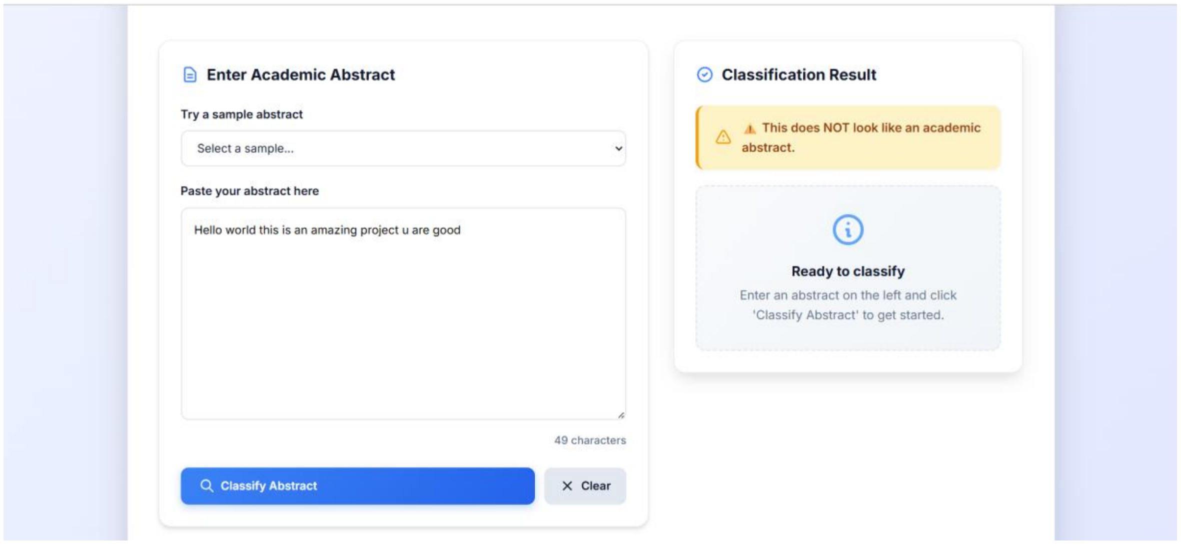
SENTIMENT ANALYSIS

NEGATIVE

Confidence Score

99.97%

8.5 Warning Message Display


Enter Academic Abstract

Try a sample abstract

Select a sample...

Paste your abstract here

Hello world this is an amazing project u are good

49 characters

Classify Abstract | Clear

Classification Result

⚠ This does NOT look like an academic abstract.

Ready to classify

Enter an abstract on the left and click 'Classify Abstract' to get started.

9.RESULTS AND DISCUSSION

Key findings:

1. Academic Validation Reliability

The validation module was able to correctly identify when the provided text did not follow academic writing style. Inputs that were too short or lacked research-oriented keywords were flagged, ensuring only meaningful abstracts proceeded to analysis.

2. Domain Detection Accuracy

The keyword-based domain classifier worked effectively for common computer science abstracts. When abstracts contained terms related to machine learning, natural language processing, or computer vision, the system matched them to the correct field without difficulty.

3. Consistent Sentiment Output from DistilBERT

DistilBERT consistently produced stable sentiment predictions across all tested samples. It successfully interpreted the tone of academic statements and represented the results with clear confidence scores.

4. Secure Access Through JWT Authentication

The JWT authentication system ensured that only logged-in users could access the main classifier page. This prevented unauthorized access and maintained a proper flow of user activity within the application.

5. Smooth and Fast Model Performance

The classifier responded quickly during testing. Since DistilBERT is lightweight, the application displayed results in real time without noticeable delays, providing an efficient user experience.

Discussion

Although the system uses a simple keyword-matching approach for domain detection, it still provides reliable results for many CS-related abstracts. For users who need quick, basic classification, this method is practical and easy to maintain.

The sentiment analysis powered by DistilBERT enhances the system by offering an interpretation of how positively or negatively the abstract communicates its findings. Even though academic writing is generally neutral, the model was able to capture slight variations in tone.

JWT authentication contributes to the overall structure by ensuring that only verified users interact with the tool. This adds a level of professionalism and security to the application.

Overall, the system combines transformer-based sentiment classification, rule-based domain detection, and a clean user interface to deliver a simple but effective abstract analysis tool. The project successfully demonstrates how modern NLP models can be integrated into a web application in a way that is accessible, fast, and useful for students and faculty.

10.LIMITATIONS

This project intentionally keeps complexity low. Therefore:

- User data is not stored in a database.
- Domain detection is not model-driven, only keyword-based.
- Sentiment classification is binary.
- Validation relies on simple keyword presence.

These limitations are natural for a student-friendly prototype.

11.FUTURE ENHANCEMENTS

- Use SciBERT or domain-specific models for academic writing.
- Expand field detection to include broader categories.
- Introduce multi-label classification.
- Implement user history using a database.
- Add PDF uploading and automatic extraction.
- Deploy on cloud platforms with HTTPS.

12.CONCLUSION

This project successfully brings together modern NLP techniques and a clean, interactive web interface to classify academic abstracts. By using DistilBERT, the system delivers fast and reliable predictions without requiring heavy computational resources. The inclusion of keyword-based field detection and academic-style validation adds practical value for users who need quick insights into research content.

Secure access through JWT authentication ensures that only verified users can interact with the classifier, giving the system a more professional and structured workflow. Overall, the project demonstrates how transformer models can be integrated into real-world applications in a simple yet effective manner. It also provides a strong foundation for expanding the tool into a more advanced academic analysis platform in the future.

13.REFERENCES

1.Vaswani, A., et al. (2017). *Attention Is All You Need.* Advances in Neural Information Processing Systems (NeurIPS).
https://arxiv.org/abs/1706.03762

2.Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). *DistilBERT, a distilled version of BERT: smaller, faster, cheaper, and lighter.*
https://arxiv.org/abs/1910.01108

3.Hugging Face Documentation. *Transformers Library.*
https://huggingface.co/docs/transformers

4.Flask Documentation. *Flask Web Framework.*
https://flask.palletsprojects.com/