LetUsSurvived_540

Master Path AI

- Kamolwan Vanichsriratana
- Syeda Raza
- Sai Naw Kham
- Htet Swan Yee

Data Science Final Project

7 Dec, 2025

LetUsSurvived_540

Motivation: Why Masterpath AI?

Explosion of academic literature

→ difficult for students to navigate

Students want to know:

"Which universities are strong in my topic?"
"How active is this field globally?"
"Who collaborates with whom?"

Existing resources require manual reading and are not personalized

Need for:

✓ Automated data processing

✓ AI topic understanding

✓ Clear visualization

✓ Personalized recommendations

LetUsSurvived_540

Project Overview

Masterpath AI consists of three major modules:

Data Module

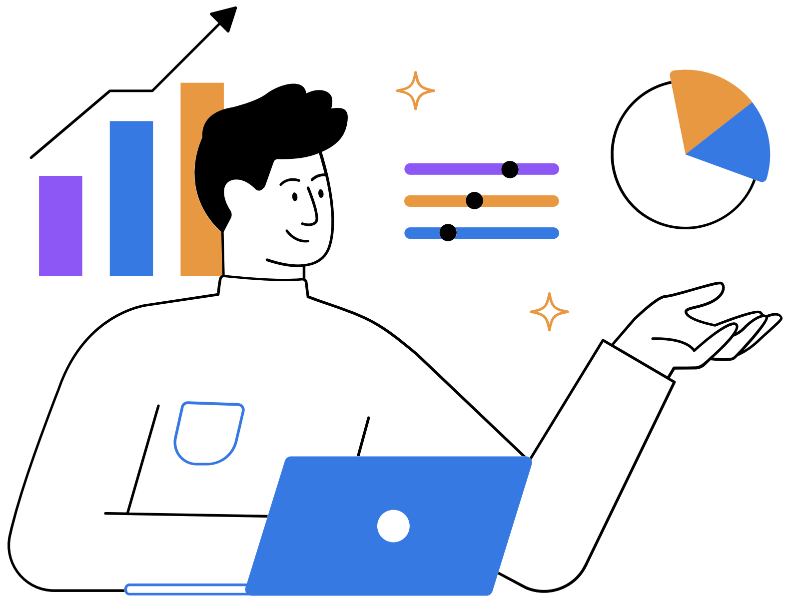
Extracts, cleans, and structures Scopus research data (2018–2023).

AI Module

Uses BERT embeddings + machine learning for topic classification and semantic recommendation

Visualization Module

Interactive Streamlit dashboard for exploration and recommendations.



```
RAW SCOPUS JSON (2018–2023)
        |
        v
┌──────────────────────────────────┐
│           DATA MODULE            │
│ – JSON Parsing                   │
│ – Cleaning & Normalization       │
│ – Institution/Country TS         │
│ – Collaboration Edges            │
└──────────────────────────────────┘
                |
        Clean Structured Data
                v
┌──────────────────────────────────┐
│            AI MODULE             │
│ – BERT Embeddings                │
│ – Topic Modeling                 │
│ – Classification                 │
│ – Recommendations                │
└──────────────────────────────────┘
                |
        Enriched AI Outputs
                v
┌──────────────────────────────────┐
│       STREAMLIT DASHBOARD        │
│ – Topic Explorer                 │
│ – Institution Explorer           │
│ – Collaboration Viewer           │
│ – Recommendation Engine          │
└──────────────────────────────────┘
```

System Architecture Diagram

LetUsSurvived_540

Data Description & Scale

Primary dataset:

Scopus Engineering research records (2018–2023)

JSON format, containing:

- Title, abstract, publication date
- Authors & affiliations
- Institution & country
- Keywords & classification codes
- Citations and references

Scale:

- 20,216 research papers
- 66,799 unique authors
- 19,334 institutions
- 176 countries

Additional data:

- country_centroids.csv for geographic visualization
- Wikipedia-based normalization of institution names

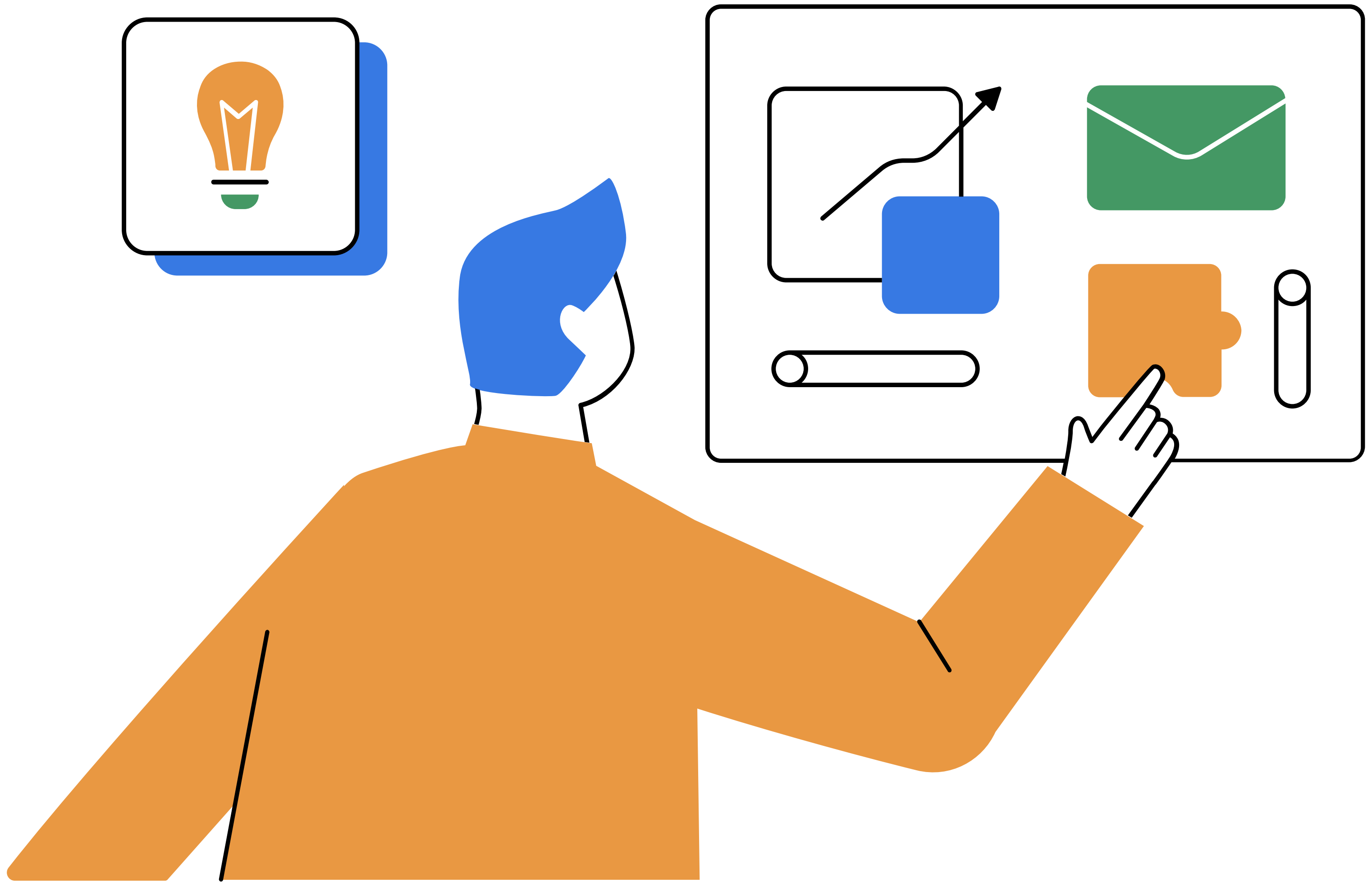
LetUsSurvived_540

DATA MODULE Overview

Data Module: Turning raw JSON into analytic data

Process:

1. Manual + automatic JSON parsing
2. Flatten nested fields
3. Normalize institutions
4. Map countries + attach geo coordinates
5. Build:
    - institution-year table
    - country-year table
    - collaboration edges
    - master dataset
6. Export cleaned CSVs



LetUsSurvived_540

```python
# IMPORTANT CHANGE: OPEN *
for file in input_dir.rglo
    if file.is_dir():
        continue

    try:
        with open(file, "r", encoding="utf-8") as f:
            data = json.load(f)  # will fail if not JSON

        files_opened += 1

        # --- Core data ---
        cores = find_all_keys(data, "coredata")
        if not cores:
            files_skipped += 1
            continue
        core = cores[0]

        paper_id = extract_paper_id(core)
        paper_title = extract_title(core)
        paper_date = extract_date(core)

        # --- Author groups ---
        author_groups = find_all_keys(data, "author-group")
        if not author_groups:
            files_skipped += 1
            continue

        for group_block in author_groups:
            groups = normalize_list(group_block)

            for group in groups:
                aff_info = extract_affiliation_info(group)

                authors = normalize_list(group.get("author"))

                for auth in authors:
                    author_name = (
                        auth.get("ce:indexed-name")
                        or auth.get("authname")
                        or "Unknown Name"
                    )
```

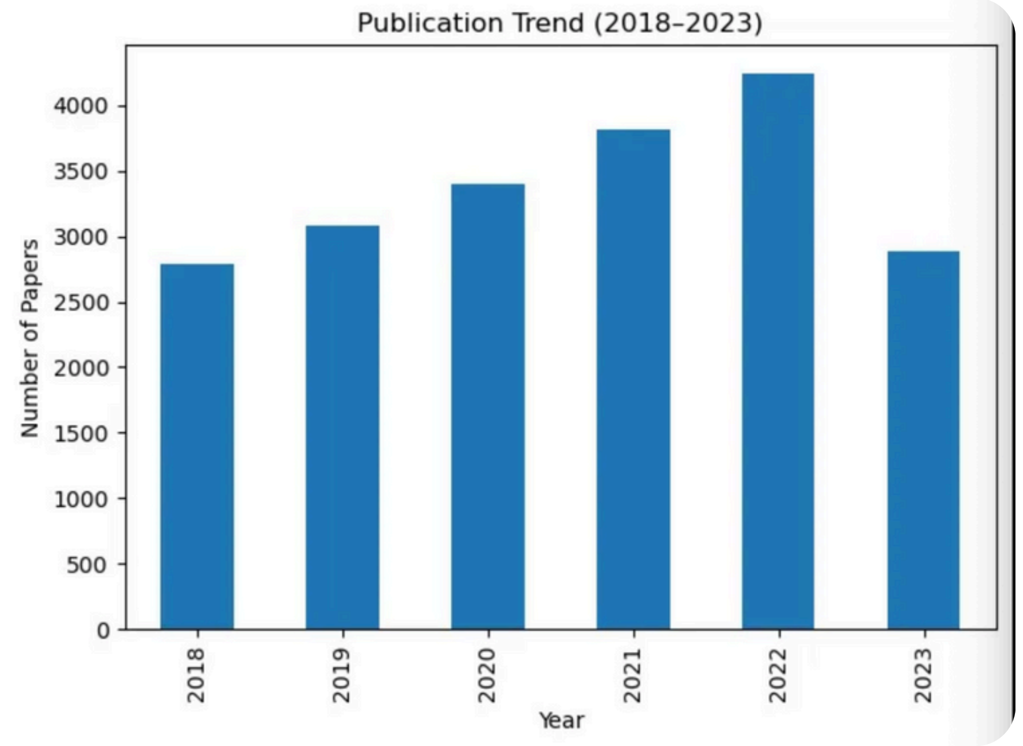
Step 1 — Load JSON files

- Loop through 20k+ raw publications
- Extract title, abstract, authors, institution, keywords

```python
for file in FILES_TO_CLEAN:
    path = DATA_DIR / file
    if not path.exists():
        print(f"Skipped {file} (not found)")
        continue

    df = pd.read_csv(path)

    # Detect institution-related columns
    inst_cols = [
        c for c in df.columns
        if "institution" in c.lower() or c in ["source", "target"]
    ]

    for col in inst_cols:
        # Standardize text
        df[col] = (
            df[col]
            .astype(str)
            .str.strip()
            .str.replace(bad_symbols_pattern, "", regex=True)
            .str.replace(r"\s+", " ", regex=True)
        )

        # Remove address-like / hospital / department / numeric garbage
        mask = df[col].str.contains(combined_pattern, na=False)
        removed = df.loc[mask, col].nunique()
        df = df[~mask]

        print(f"{file}: removed {removed} invalid entries from '{col}'")

        # Remove empty & ultra-short garbage values
        df = df[df[col].notna()]
        df = df[df[col].str.len() >= 4]
```

Step 2 — Clean & normalize institution names

- Remove punctuation
- Apply canonical mapping
- Uniform institution identities
  (MIT, Massachusetts Institute of Technology → MIT)



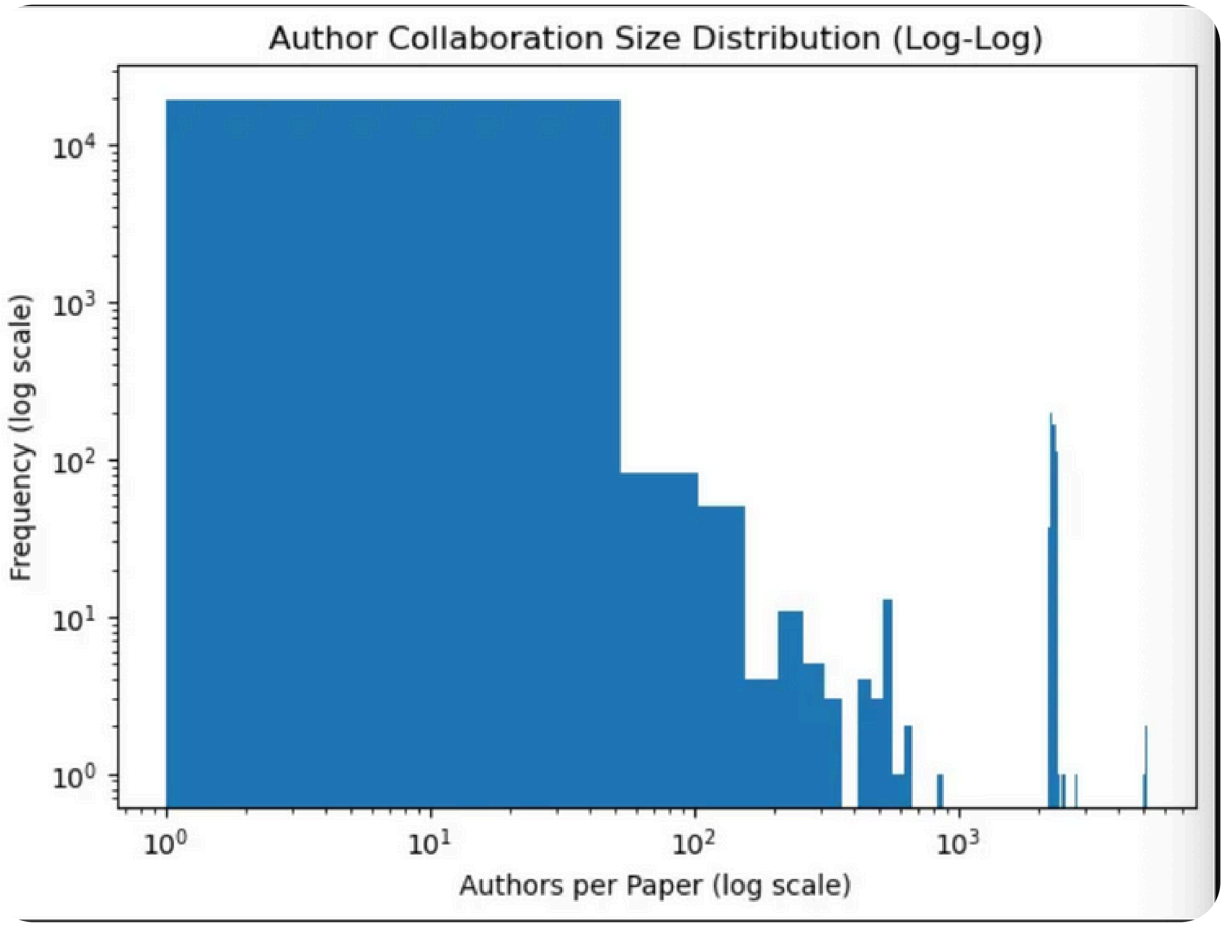
Step 3 — Build institution-year statistics

- Count papers per year
- Used for trend visualization

Step-by-Step: Data Module (Deep Dive)

LetUsSurvived_540

Step-by-Step: Data Module (Deep Dive)




```
institution_year_output.csv: removed 0 invalid entries from 'institution'
Fully cleaned: institution_year_output.csv
institution_collaboration_edges.csv: removed 0 invalid entries from 'source'
institution_collaboration_edges.csv: removed 0 invalid entries from 'target'
Fully cleaned: institution_collaboration_edges.csv
institution_ai_summary.csv: removed 0 invalid entries from 'institution'
Fully cleaned: institution_ai_summary.csv
MASTER CLEANING COMPLETE — only true university institutions remain.
```

```
Cleaned: institution_ai_summary.csv
Cleaned: institution_year_output.csv
Cleaned: institution_collaboration_edges.csv
All ghost institutions removed.
```

Step 4 — Build collaboration network

- For each paper: generate pairs of institutions
- Count collaboration frequency
  → edge weights

➔

Step 5 — Merge geographic coordinates

- Join with country_centroids.csv
- Required for pydeck map visualization

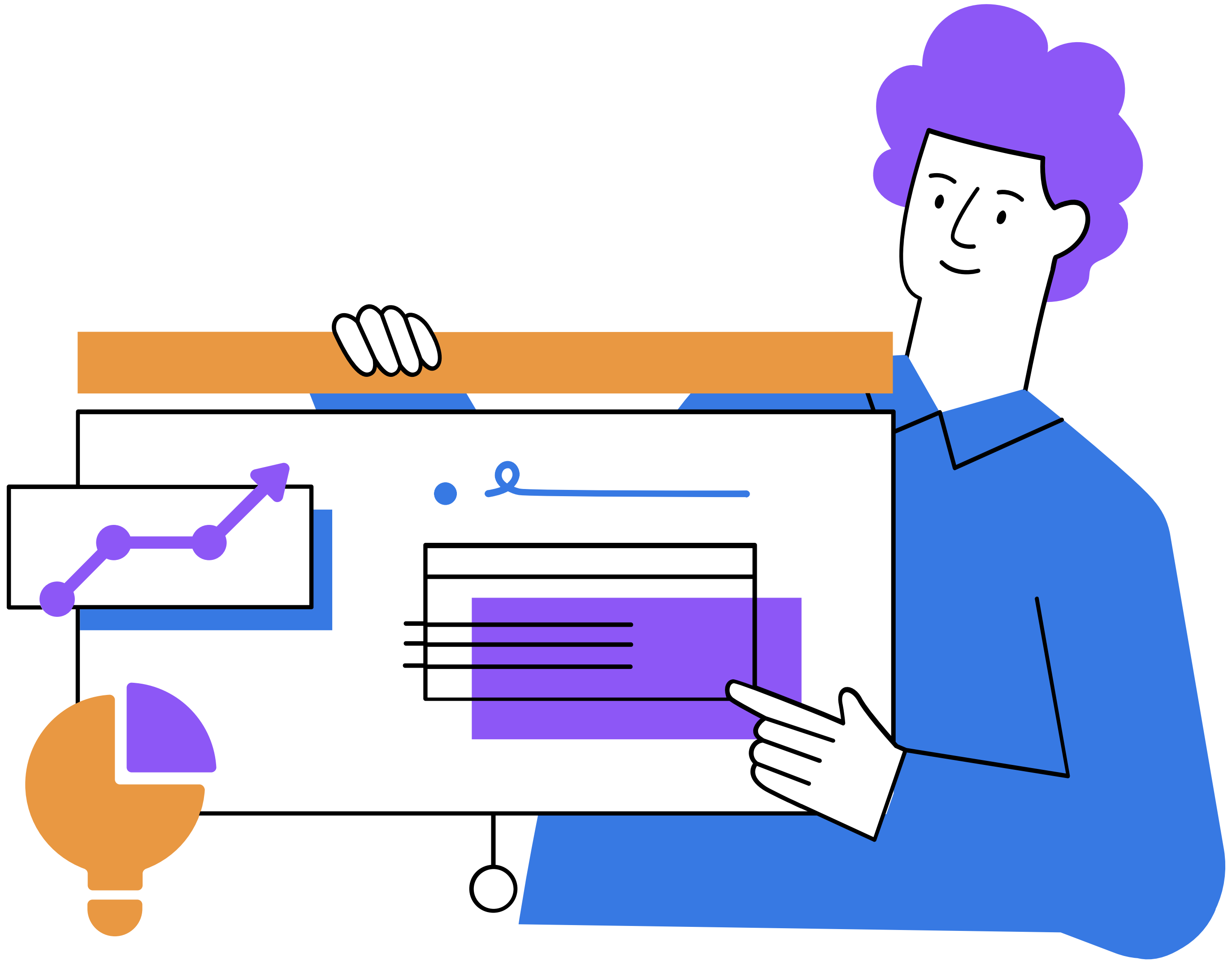
LetUsSurvived_540

AI MODULE Overview

AI Module: Understanding Topics & Making Predictions

Components:

1. Text preprocessing
2. Semantic embedding with Sentence-BERT
3. Topic clustering
4. Supervised classification (Logistic Regression)
5. Evaluation metrics
6. University recommendation engine



LetUsSurvived_540

Step-by-Step: AI Module (Deep Dive)

```
inst_year = pd.read_csv(DATA_DIR / "institution_year_output.csv")
edges = pd.read_csv(DATA_DIR / "institution_collaboration_edges.csv")

print("institution_year_output:", inst_year.shape)
print(inst_year.head())

print("\ncollaboration_edges:", edges.shape)
print(edges.head())
```

```
institution_year_output: (13518, 4)
                       institution  year  num_papers  author_count
0                 Luigi Vanvitelli  2021           1             3
1  Magna Graecia University of Catanzaro  2022           1             1
2               Mater Domini Hopsital  2023           1             1
3                              CEP  2021           1             1
4                     IRyCIS Madrid  2023           1             1

collaboration_edges: (485366, 3)
                    source                  target  weight
0   Chulalongkorn University      Mahidol University     976
1         Mahidol University  Chulalongkorn University     801
2           Peking University    University of Hamburg     530
3  Université Libre de Bruxelles   University of Hamburg     521
4  Université Libre de Bruxelles       Peking University     521
```

```
# ==========================================
# 7. SAVE ALL BERT EMBEDDINGS (FOR FUTURE USE)
# ==========================================

print("Encoding ALL institutions for embedding export...")
all_embeddings = bert.encode(df["text"].values, show_progress_bar=True)

emb_df = pd.DataFrame(all_embeddings)
emb_df["institution"] = df["institution"].values
emb_df["label"] = df["label"].values

emb_df.to_csv("data/bert_embeddings.csv", index=False)
print("BERT embeddings saved to data/bert_embeddings.csv")
```

```
Loaded: (7426, 15)
Label distribution:
label
0    5564
1    1862
Name: count, dtype: int64
Loading BERT model...
Encoding train set...
```

```
# ==============================================
# 2. TRAIN / TEST SPLIT
# ==============================================

X_train, X_test, y_train, y_test = train_test_split(
    df["text"].values,
    df["label"].values,
    test_size=0.2,
    random_state=42,
    stratify=df["label"]
)
```

Step 1 — Load processed data

- Clean text from Data Module
- Combine title + abstract

→

Step 2 — Generate BERT embeddings

- 768-dim feature vectors
- Each vector = semantic meaning of a paper

→

Step 3 — Split train/test

- Maintain label distribution
- Prevent overfitting

LetUsSurvived_540

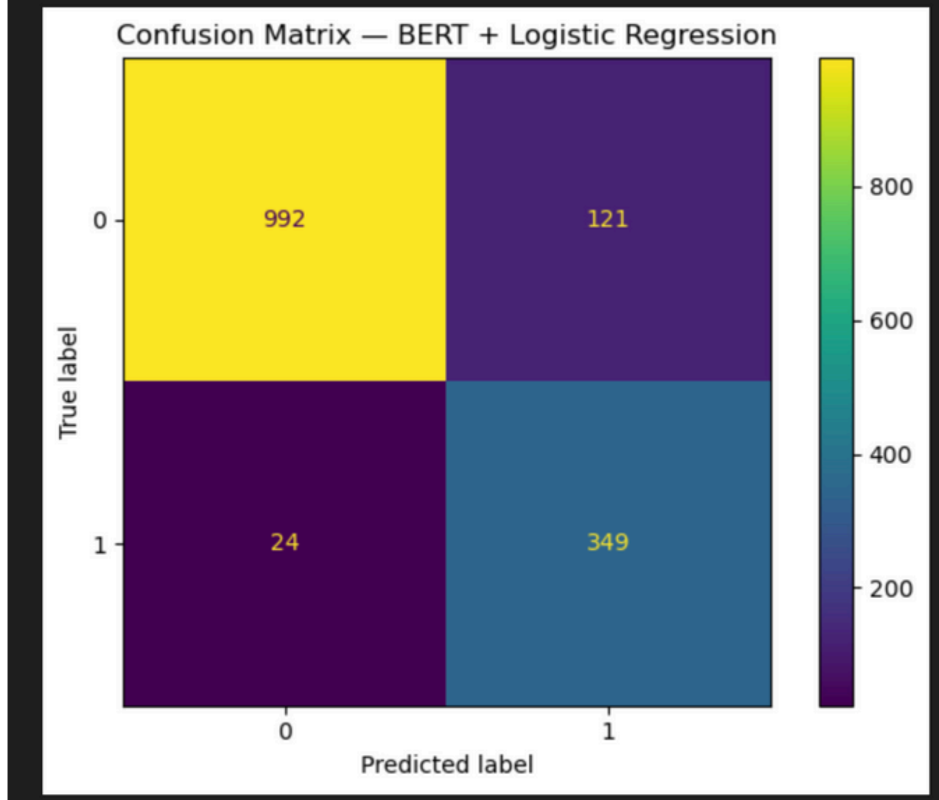
Step-by-Step: AI Module (Deep Dive)

```python
# ==================
# BERT + CLASSIFIER
# ==================

from sentence_transformers import SentenceTransformer
from sklearn.metrics import (
    accuracy_score, precision_score, recall_score,
    f1_score, roc_auc_score, classification_report
)

# ============================================
# 1. LOAD YOUR EXISTING AI SUMMARY FILE
# ============================================

DATA_PATH = "data/institution_ai_summary.csv"
df = pd.read_csv(DATA_PATH)

print("Loaded:", df.shape)

# --------------------------------------------
# Create TEXT field for BERT
# We combine institution name + numeric signals as text
# --------------------------------------------

df["text"] = (
    df["institution"].astype(str) + " " +
    "cluster " + df["cluster"].astype(str) + " " +
    "degree " + df["weighted_degree"].astype(str)
)

# --------------------------------------------
# Create LABEL automatically (Top vs Non-Top)
# Using top 25% of weighted_degree
# --------------------------------------------

threshold = df["weighted_degree"].quantile(0.75)
```

Step 4 — Train classifier

- Logistic Regression on BERT embeddings
- Predicts topic or AI category

```
<Figure size 640x480 with 0 Axes>
```




```
Confusion matrix saved to data/confusion_matrix_bert.png
```

Step 5 — Evaluate

- Accuracy, precision, recall, F1
- Save in ai_model_metrics.csv

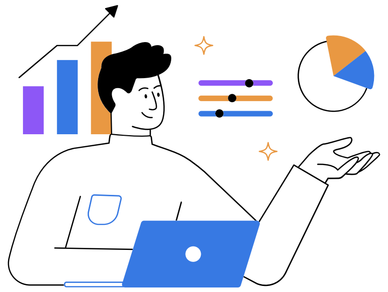
2. Build institution feature matrix (for clustering & classification)

```python
# Ensure year is int
inst_year["year"] = inst_year["year"].astype(int)

# Pivot: one row per institution, columns = year's paper count
year_cols = sorted(inst_year["year"].unique())
inst_pivot = (
    inst_year
    .pivot_table(
        index="institution",
        columns="year",
        values="num_papers",
        aggfunc="sum",
        fill_value=0
    )
)

inst_pivot = inst_pivot.reset_index()
inst_pivot.columns.name = None

print("Pivoted institution features:", inst_pivot.shape)

feature_cols = [c for c in inst_pivot.columns if isinstance(c, (int, np.integer))]
print("Feature columns (years):", feature_cols)
```

Step 6 — Topic modeling

- Cluster embeddings into topics
- Extract keywords for interpretation

LetUsSurvived_540



AI Module: Recommendation Engine

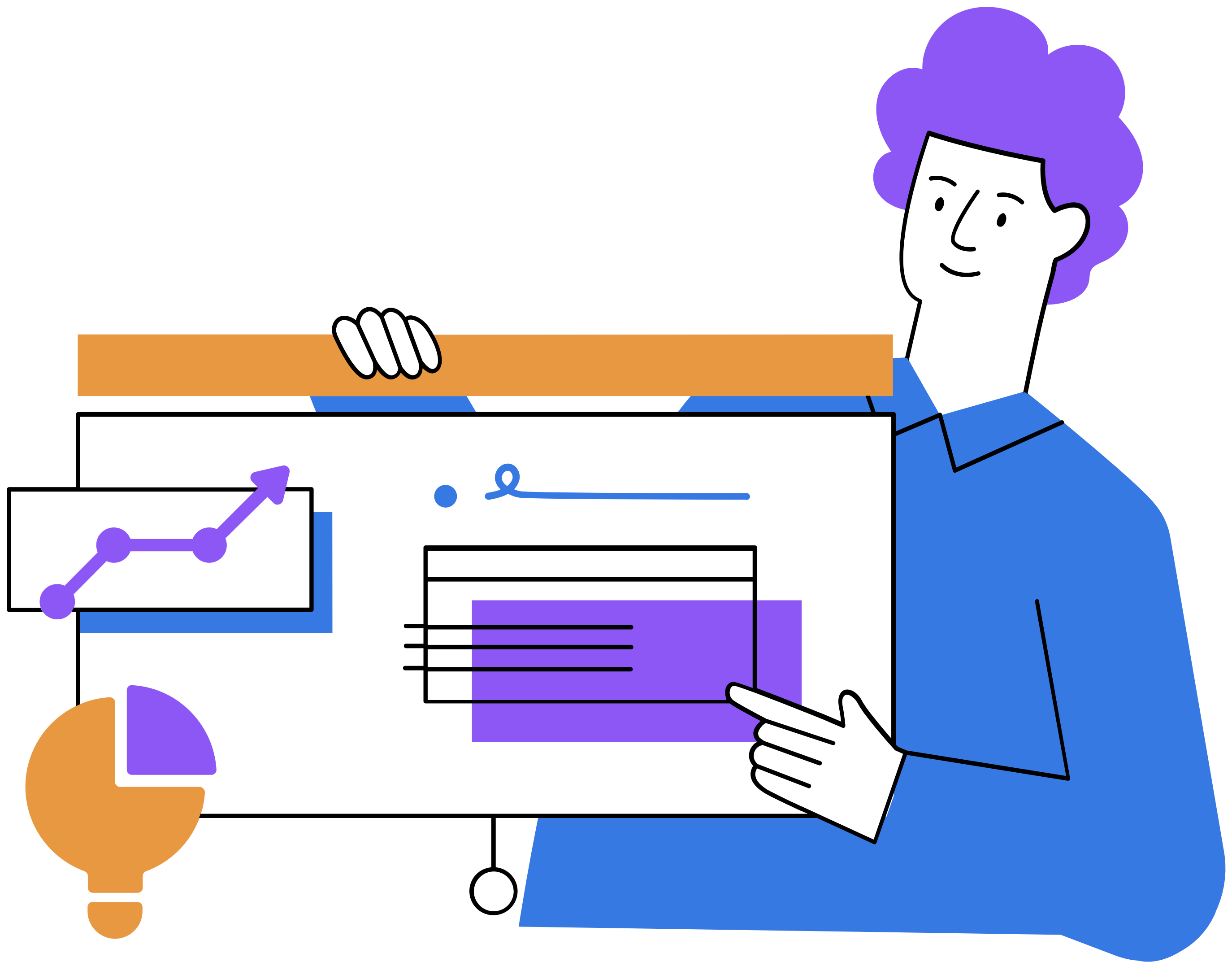
Example Output:

Rank	University	Match Score	Strength
1	MIT	0.93	Deep Learning
2	Tsinghua	0.91	Robotics/ML
3	U of Toronto	0.89	Optimization/ML

How the recommendation works:

1. Student enters their research interest
2. Model encodes the text into a BERT embedding
3. System compares embedding to institutional topic profiles
4. Ranks universities using:
   - Topic similarity
   - AI intensity score
   - Number of relevant publications
5. Produces Top-N recommended universities

LetUsSurvived_540



Visualization Module

Streamlit Dashboard:

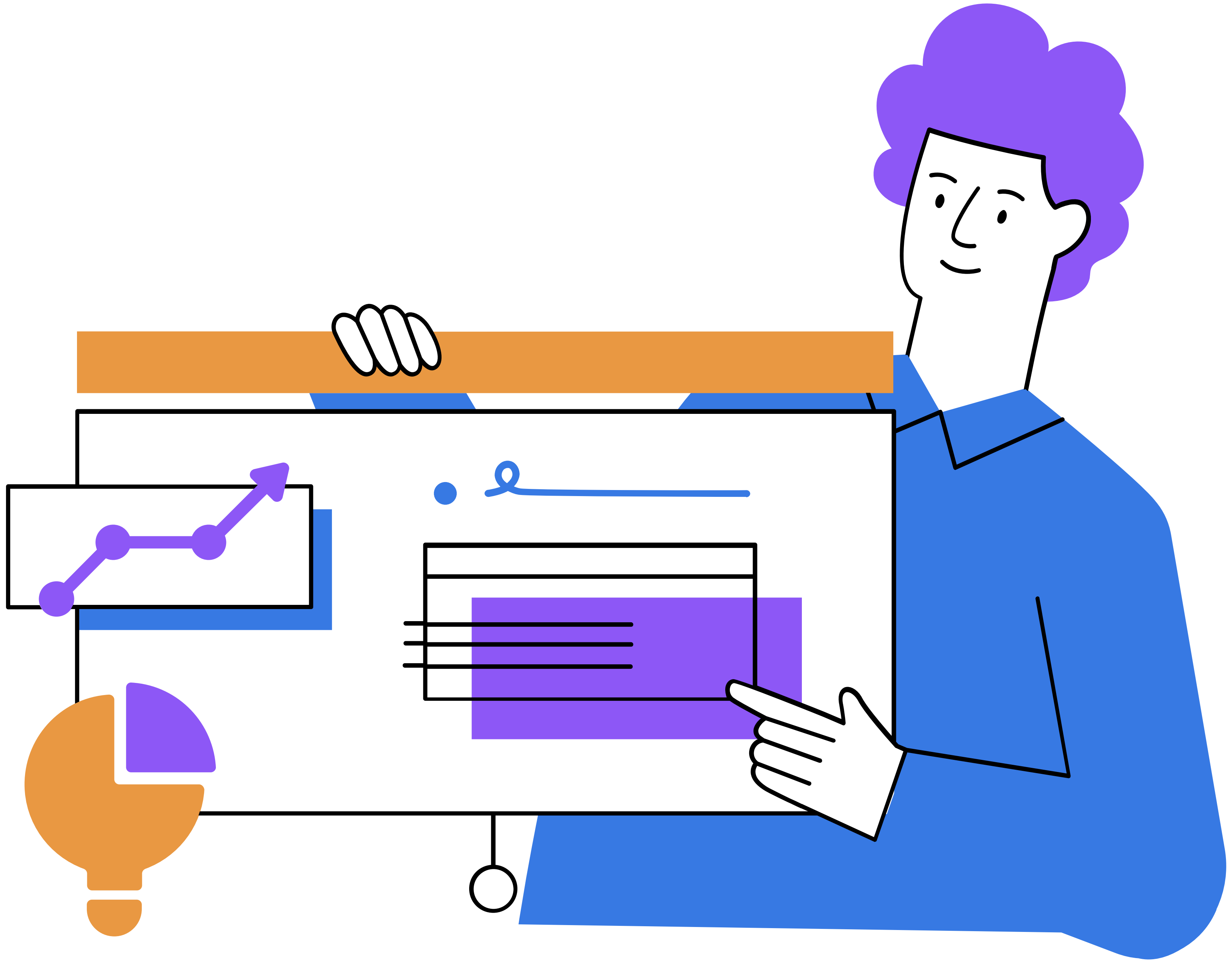
Making Insights Interactive

Pages:

1. Home Overview
2. Topic Explorer
3. Institution Explorer
4. Collaboration Network
5. Search & Recommendation
6. AI Model Performance

LetUsSurvived_540

Institution Explorer Demo



Content:

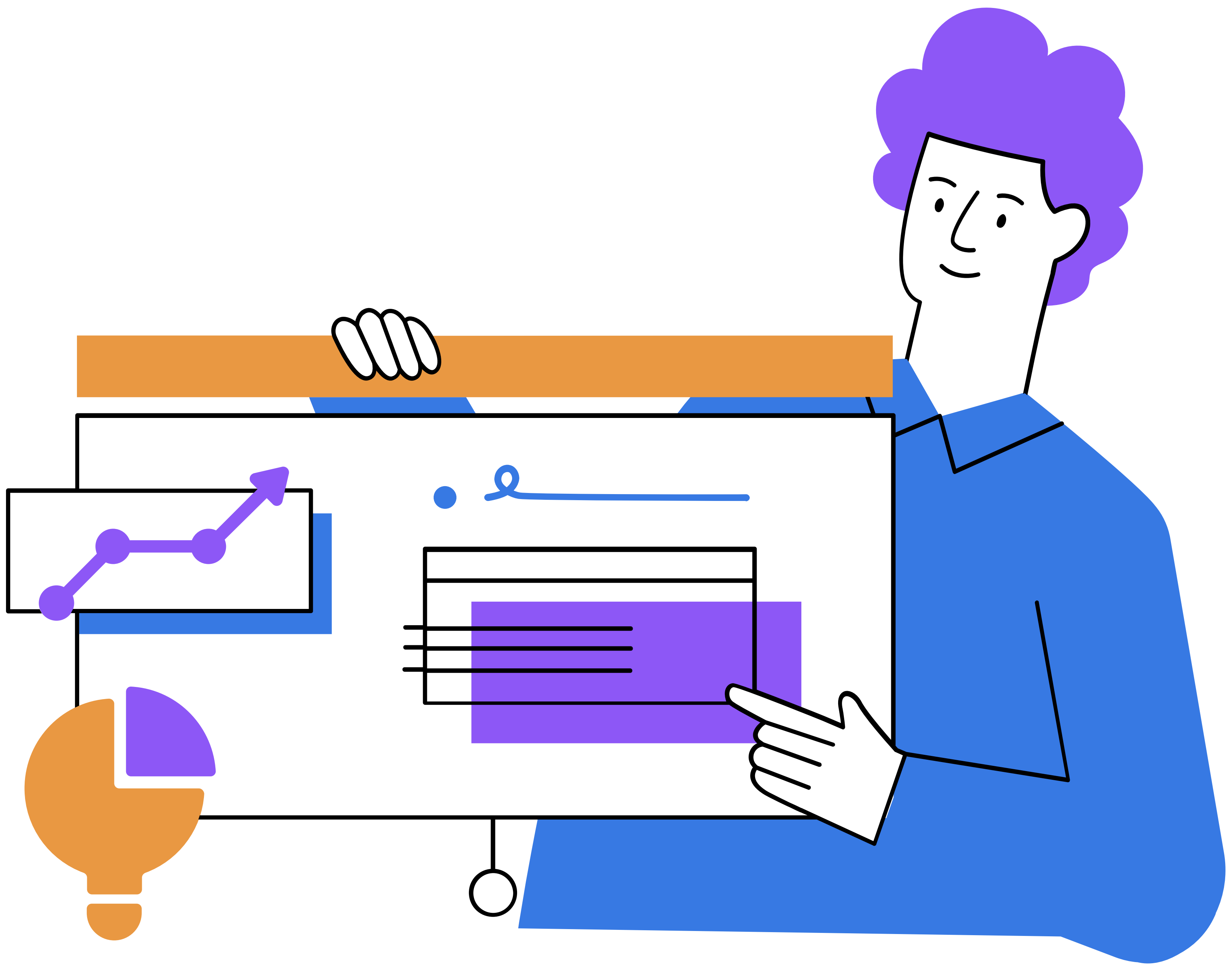
- Select an institution
- See:
    - Paper count per year
    - Topic dominance
    - Overall AI intensity
- Compare institutions

Collaboration Network Demo

Content:

- Visual network of institutional collaborations
- Node size = research volume
- Edge width = collaboration strength

LetUsSurvived_540



Recommendation Page Demo

Content:

- User inputs research interest
- Dashboard outputs top recommended universities
- Built from AI module's semantic engine

LetUsSurvived_540

Technical Achievements

- Processed large-scale JSON data (20k+ publications)
- Built an institution-level research analytics pipeline
- Applied BERT embeddings for semantic understanding
- Built a topic-based classifier
- Generated AI evaluation metrics automatically
- Designed a multi-page Streamlit dashboard
- Implemented real-time university recommendation

LetUsSurvived_540

Challenges:

- Messy Scopus JSON
- Ambiguous institution names
- High-dimensional embedding space
- Slow processing

Solutions:

- Recursive JSON parsing
- Institution name normalization
- Dimensionality reduction / batching
- Cached loading in Streamlit

Challenges & Solutions

LetUsSurvived_540

Conclusion

Masterpath AI provides:

✓ A complete research analytics ecosystem

✓ Smart topic-based university recommendations

✓ Interactive visual exploration

✓ AI-powered insights for student decision-making

LetUsSurvived_540

Future Enhancements

Add citation-based impact scores

Integrate Google Scholar or Dimensions API

Add GPT topic summarization

Deploy online for public use

Add comparison mode between two universities

LetUsSurvived_540

Master Path AI

- Kamolwan Vanichsriratana
- Syeda Raza
- Sai Naw Kham
- Htet Swan Yee

Data Science Final Project

7 Dec, 2025