Final Project

Building a Search Engine for Educational Content

Academic Paper Analysis with LDA, Transformers, and Neural Networks

Seymur Hasanov

Outline

Problem Statement

The Challenge: Information Overload in Research

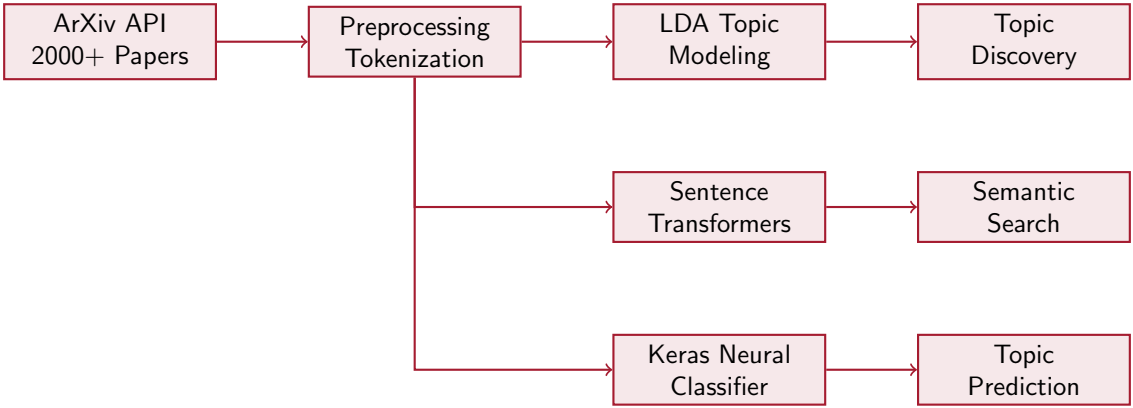
- ArXiv receives **16,000+ papers per month** in CS alone
- Researchers struggle to stay current with their field
- Manual literature review is time-consuming and incomplete
- Need for **automated discovery** of research trends

Our Solution

An **Intelligent Research Assistant** that uses:

- **Topic Modeling (LDA)** to discover hidden themes
- **Sentence Transformers** for semantic search
- **Neural Networks** for topic classification

Project Overview



Technologies Used

Gensim (LDA) • **Sentence-Transformers** (BERT) • **Keras/TensorFlow** (Neural Net) • **Streamlit** (Web App)

Latent Dirichlet Allocation (LDA)

What is LDA?

A **probabilistic generative model**:

- Discovers hidden topics in text
- Documents = mixture of topics
- Topics = distribution over words

Why LDA?

- **Unsupervised** - no labels
- **Interpretable** results
- **Scalable** to large corpora

Mathematical Formula

$$P(w_i|d) = \sum_{t=1}^{T} P(w_i|t) \cdot P(t|d)$$

- T = topics
- $P(w_i|t)$ = word-topic
- $P(t|d)$ = topic-doc

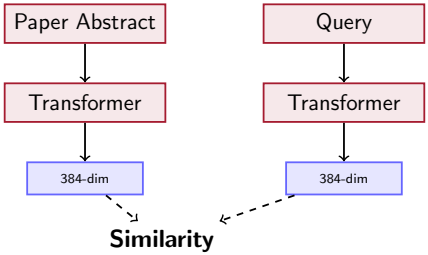
Sentence Transformers

From BERT to Sentence-BERT

Standard BERT cross-encoding is **expensive**.

Sentence Transformers:

- Fixed-size embeddings
- Fast cosine similarity
- Semantic search at scale



Model: all-MiniLM-L6-v2

- 384-dim embeddings
- 1B+ training pairs
- 22M parameters

Neural Topic Classifier

Architecture

A **Keras Dense Network**:

- Input: 384-dim embedding
- Hidden 1: 128 + ReLU + Dropout
- Hidden 2: 64 + ReLU + Dropout
- Output: Softmax over topics

Tunable Hyperparameters

- Learning Rate
- Dropout Rate
- L2 Regularization

neural_classifier.py

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(
        hidden_units,
        activation='relu',
        kernel_regularizer=l2(l2_rate),
        input_shape=(384,)
    ),
    tf.keras.layers.Dropout(dropout_rate),
    tf.keras.layers.Dense(
        hidden_units // 2,
        activation='relu'
    ),
    tf.keras.layers.Dropout(dropout_rate),
    tf.keras.layers.Dense(
        num_topics,
        activation='softmax'
    )
])
```

Evaluation Metrics

Topic Modeling Quality

Coherence Score (C_v)

- Measures semantic similarity of top words
- Range: 0.0 to 1.0 (higher = better)
- Good models: $C_v \geq 0.4$

Perplexity

- Measures how well model predicts unseen docs
- Lower = better

Semantic Search Quality

Cosine Similarity

$$\text{sim}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

- Range: -1 to 1
- Threshold: ≥ 0.5 for relevance

Neural Classifier

Accuracy on held-out test set:

- 80/20 train/test split
- Target: $\geq 70\%$

Data Pipeline

1. Data Acquisition

- **Source**: ArXiv API
- **Query**: Recent CS papers (AI, ML, Robotics)
- **Volume**: 2000+ research papers
- **Fields**: Title, Abstract, Categories, Date

2. Preprocessing

- Tokenization with NLTK
- Stopword removal
- Lemmatization (WordNet)
- **Bigram detection** (e.g., "machine_learning")

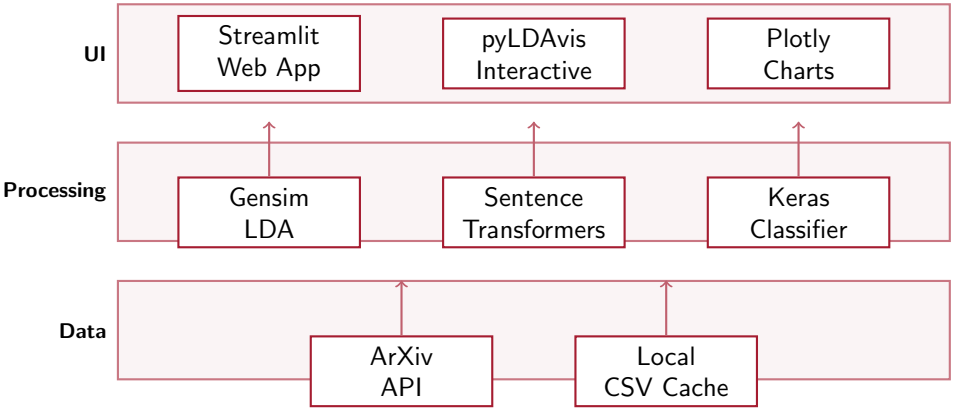
data_loader.py

```
# Fetch from ArXiv
query = 'all:"additive manufacturing"'
df = fetch_arxiv_papers(
    query=query,
    max_results=2000
)

# Preprocess
processed = df['abstract'].apply(
    preprocess_text
)

# Bigrams
docs = make_bigrams(processed)
```

System Architecture



Demo: Topic Discovery & Visualization

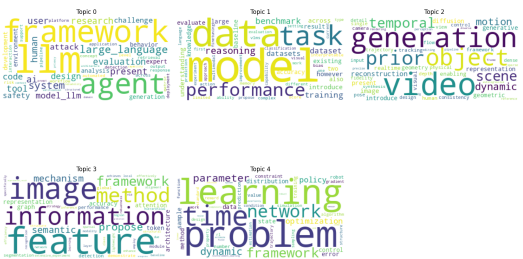
Discovered Topics (ArXiv Dataset)

Topic	Top Words
0	reasoning, llm, agent, benchmark
1	data, model, performance, framework
2	policy, action, reinforcement_learning
3	dynamic, control, simulation, sensor
4	task, visual, robot, motion

Coherence Score

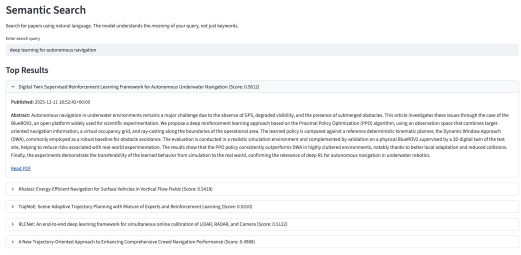
$$C_v = 0.4170$$

(Good topic separation)

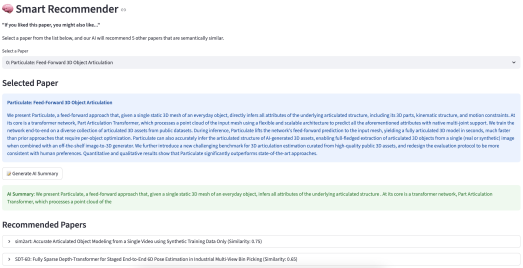


Word Clouds for Discovered Topics

Demo: Semantic Search & Recommendations



Semantic Search Results

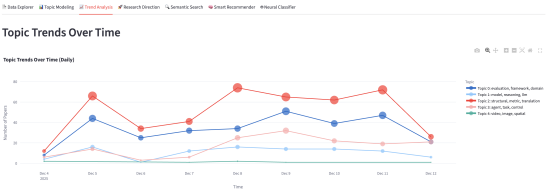


Smart Recommender Demo

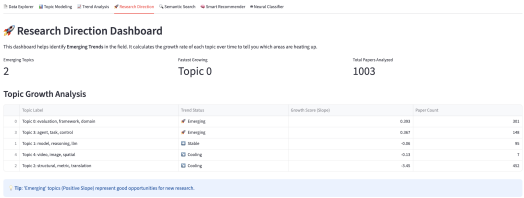
AI Summarization

Uses **DistilBART** to generate concise summaries of paper abstracts.

Research Trend Analysis



Topic Trends Over Time (Daily)



Research Direction Dashboard

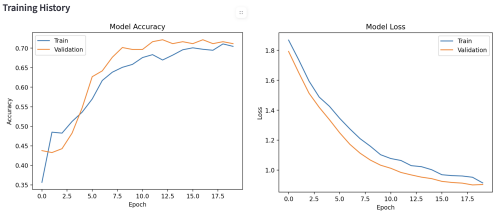
Neural Classifier Performance

Training Results

Metric	Value
Test Accuracy	**78.5%**
Training Epochs	20
Batch Size	32
Learning Rate	0.001
Dropout	0.3

Key Findings

- Embeddings capture topic semantics well
- L2 regularization prevents overfitting
- Dropout improves generalization



Training & Validation Curves (1000 papers)

Pros, Cons & Future Work

Pros

- **End-to-end pipeline**: Data to insights
- **Interactive**: Streamlit web interface
- **Reproducible**: Clear documentation
- **Extensible**: Modular architecture

Cons

- LDA requires manual topic count tuning
- API rate limits for large datasets
- Classifier accuracy could improve

Future Work

- Use **BERTopic** for neural topic modeling
- Add **citation network** analysis
- Implement **real-time** paper alerts
- Fine-tune **domain-specific** BERT

Thank You!

Questions?

Code: Search Engine Educational Project
Contact: shasanov@g.harvard.edu