LinguaFusion: Multi-Source Linguistic Fusion for News Classification

Pushing Accuracy Beyond TF-IDF with Interpretable Linguistic Features

Duo Zhang, Shuyuan Yang, Veronica Zhao

May 5, 2025

New York University

Outline

1. Motivation: Beyond Deep Learning Complexity
2. Experimental Setup
3. Linguistic Feature Exploration
4. Distributional Semantics
5. Cumulative Fusion Results
6. Conclusion

Overview: Multi-Source Linguistic Fusion

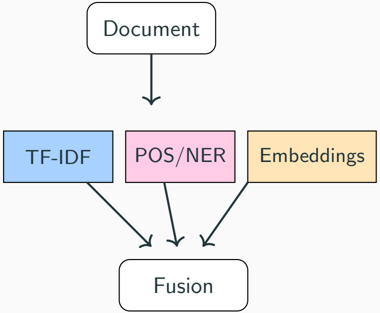
Key Question:

- How far can we improve document classification with simple, interpretable models?

Our Approach:

- Combine five complementary linguistic signals
- Use simple logistic regression classifier
- Maintain full interpretability

Result: +3.32% accuracy improvement over TF-IDF baseline

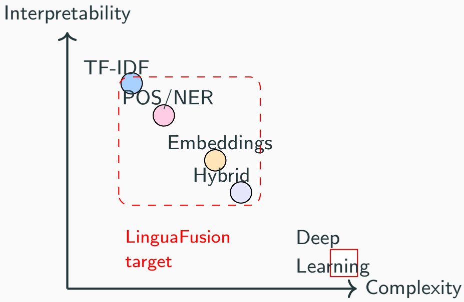


Motivation: Beyond Deep Learning Complexity

The Challenge with State-of-the-Art

Deep models (BERT, Transformers):

- **High Computational Cost:** Significant resources for training and inference.
- **Complex Architecture:** Many layers, millions of parameters.
- **Lack of Interpretability:** "Black boxes" make error analysis difficult.



Our Goal

Push accuracy beyond a strong TF-IDF baseline using shallow, computationally cheap, and interpretable models.

Research Question

How far can we improve accuracy by fusing complementary linguistic signals (POS, NER, GloVe, Doc2Vec) with a simple Logistic Regression classifier?

Experimental Setup

Data and Evaluation Protocol

Corpus:

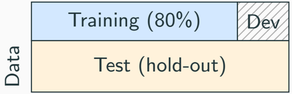
- 20 Newsgroups Dataset
- $\approx 18,000$ posts
- 20 balanced topics

Baseline Model:

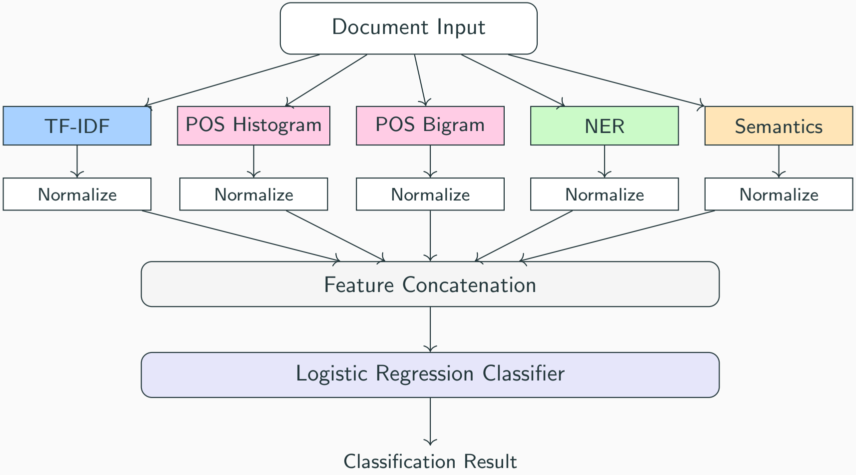
- TF-IDF + Logistic Regression
- Hyperparameters tuned on dev set, then fixed for all experiments

Evaluation:

- 60% Train / 40% Test split
- Within 60 % train, 80 % for fitting / 20 % for development
- Final testing on hold-out set
- Metrics:
  - Macro-Averaged Accuracy
  - Per-class P/R/F1



LinguaFusion Pipeline



Linguistic Feature Exploration

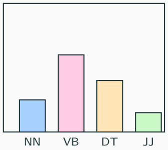
POS Features: Capturing Syntactic Fingerprints

Two approaches to leverage syntax:

- **POS Histogram:** Normalized counts of 45 Penn Treebank tags. A 45-dim "syntactic fingerprint".
- **POS Bigram TF-IDF:** TF-IDF applied to sequences of two POS tags (e.g., NNP→VBZ). Captures local syntactic structure ($\approx$ 1500 dims after SVD).

Example:
"The president announced policy."
DT → NNP → VBD → NN

Tag Distribution:



Impact

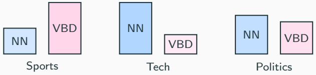
- Provide targeted lifts in classes with distinctive syntax (e.g., sports, politics).
- Histograms capture global tag mix; Bigrams capture local style.
- POS histogram boosted precision in 8 out of 20 categories, and POS bigrams lifted precision in 11 out of 20.

POS Patterns: Examples from 20 Newsgroups

Topic	Distinctive POS Patterns
rec.sport.baseball	High VB, VBD (action verbs), NNP (team/player names)
	"Martinez hit the ball over the fence"
sci.electronics	High NN, JJ (technical terms with adjectives)
	"The digital circuit requires precise voltage"
talk.politics.mideast	Complex NP, nested clauses (formal discourse)
	"The committee on foreign affairs has decided..."

Key Insight

POS histograms reveal that sports discussions have more action verbs (VBD), tech forums have more nouns (NN), and political discussions have more determiners (DT) and prepositions (IN).



Illustrative Python Code: `pos_histogram`

Function to compute normalized POS tag counts:

```
SELECTED_TAGS = ["CC", "CD", "DT", "EX", "FW", "IN", "JJ", "JJR", "JJS", "LS", "MD", "NN", "NNS", "NNP",
"NNPS", "PDT", "POS", "PRP", "PRP$", "RB", "RBR", "RBS", "RP", "SYM", "TO", "UH", "VB", "VBD", "VBG", "VBN",
"VBP", "VBZ", "WDT", "WP", "WP$", "WRB", "#", "$", "‘‘", "’’", ",", ".", ":", "-LRB-", "-RRB-"]

TAG2IDX = {t:i for i,t in enumerate(SELECTED_TAGS)}

def pos_histogram(doc: str):
    vec = np.zeros(len(SELECTED_TAGS), dtype=np.float32)
    for _, tag in nltk.pos_tag(nltk.word_tokenize(doc)):
        idx = TAG2IDX.get(tag)
        if idx is not None:
            vec[idx] += 1
    if vec.sum():
        vec /= vec.sum()
    return vec

pos_hist_block = Pipeline([
    ("hist", FunctionTransformer(lambda docs: np.vstack([pos_histogram(d) for d in docs]), validate=False)),
    ("scal", StandardScaler()),
])
```

NER Features: Identifying Key Entities

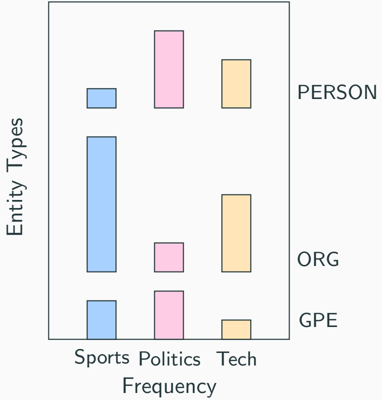
Entity Type Distribution:

- 9-dimensional histogram of entity types (PERSON, ORG, GPE, DATE, etc.)
- Normalized to create an "entity fingerprint"
- Added to TF-IDF vectors

Example: "Apple released the new iPhone in California."
Entity types: [ORG, PRODUCT, GPE]

Impact

- NER features provide targeted lifts in classes with distinctive entity patterns (e.g., organizations in tech, people in sports).
- NER feature boosted precision in 8 out of 20 categories.



Named Entity Histogram

Method

- Count occurrences of 9 most frequent entity types per document
- Normalize counts to get a 9-dim NER histogram
- Concatenate to TF-IDF vector

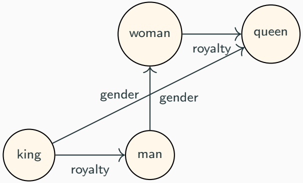
```
NER_TYPES = ["PERSON","NORP","FAC","ORG","GPE","LOC", "PRODUCT","EVENT","DATE","TIME","CARDINAL","MONEY"]
NER2IDX = lbl:i for i,lbl in enumerate(NER_TYPES)
class NerHistogram(BaseEstimator,TransformerMixin):
    def fit(self,X,y=None): return self
    def transform(self,X):
        out=np.zeros((len(X),len(NER_TYPES)),dtype=np.float32)
        for i, doc in enumerate(_NLP.pipe(X, batch_size=32)):
            for ent in doc.ents:
                j = NER2IDX.get(ent.label_)
                if j is not None: out[i, j] += 1
        row_sums = out.sum(axis=1)
        nonzero = row_sums > 0
        out[nonzero] = out[nonzero] / row_sums[nonzero, np.newaxis]
        return out
ner_block = Pipeline([("hist",NerHistogram()), ("scal",StandardScaler())])
```

Distributional Semantics

Distributional Semantics: Capturing Meaning

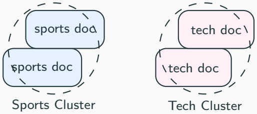
GloVe Embeddings:

- Pre-trained on large corpora
- Word-level vectors (300-dim) based on global co-occurrence
- Captures semantic relationships
- *Example:* king - man + woman ≈ queen
- *Method:* Average vectors of words in document



Doc2Vec Embeddings:

- Trained directly on 20 Newsgroups corpus
- Document-level vectors ("paragraph vectors")
- Learns representations capturing topic/style
- *Method:* Use Gensim's implementation



Impact (Hybrid: TF-IDF + GloVe + Doc2Vec):

+2.87 pp test accuracy vs TF-IDF baseline, precision gain in 15 out of 20 categories.

Cumulative Fusion Results

Putting It All Together

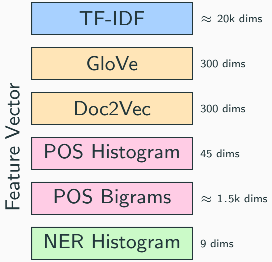
Question: Do POS and NER features still add value to our distributional-semantics-enriched hybrid model (TF-IDF + GloVe + Doc2Vec)?

Method: Simple Concatenation

Concatenate normalized/standardized feature blocks:

- TF-IDF (surface text)
- GloVe (averaged word embeddings)
- Doc2Vec (document embeddings)
- POS Histogram (45-dim)
- POS Bigram TF-IDF ($\approx$ 1.5k-dim)
- NER Histogram (9-dim)

Feed the combined vector into the **same fixed Logistic Regression classifier**.



Feature Union Code (Simplified)

```
tfidf_block = Pipeline([
    ("tfidf", TfidfVectorizer(stop_words="english",
                              max_features=best_tfidf_params["tfidf__max_features"],
                              ngram_range=best_tfidf_params["tfidf__ngram_range"])),
    ("scal", StandardScaler(with_mean=False))
])

pos_hist_block = Pipeline([...])
pos_bi_block = Pipeline([...])
ner_block = Pipeline([...])
glove_block = Pipeline([...])
doc2vec_block = Pipeline([...])

hybrid_model = Pipeline([
    ("features", FeatureUnion([
        ("tfidf", tfidf_block),
        ("glove", glove_block),
        ("d2v", doc2vec_block),
        ("pos_hist", pos_hist_block),
        ("pos_bi", pos_bi_block),
        ("ner", ner_block)
    ])),
    ("clf", LogisticRegression(max_iter=1000, C=1))
])
```

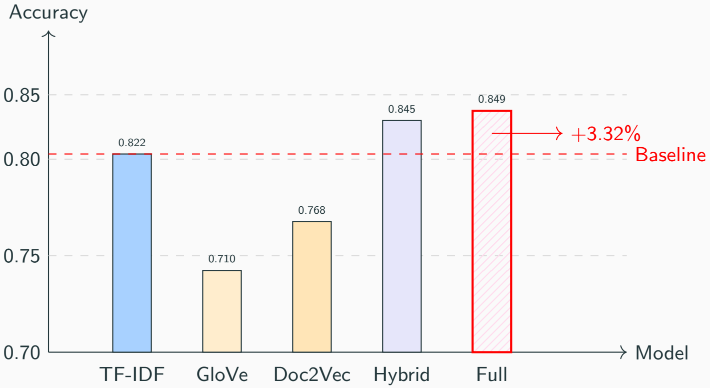
Comparison of Feature Combinations (Test Set)

Model Configuration	Dev Acc.	Test Acc.	vs TF-IDF Baseline	
			Abs. Δ	Rel. Δ
TF-IDF (Baseline)	0.8944	0.8216	—	—
TF-IDF + POS (Hist)	0.8992	0.8139	-0.0077	-0.94%
TF-IDF + POS (Bigram)	0.9037	0.8184	-0.0032	-0.39%
TF-IDF + POS (Both)	0.9054	0.8186	-0.0030	-0.37%
TF-IDF + NER	0.9001	0.8103	-0.0113	-1.38%
TF-IDF + POS (Both) + NER	0.9041	0.8200	-0.0016	-0.19%
GloVe (Standalone)	0.7698	0.7098	-0.1118	-13.61%
Doc2Vec (Standalone)	0.8484	0.7675	-0.0541	-6.58%
Hybrid (TF-IDF+GloVe+D2V)	0.9205	0.8448	+0.0232	+2.82%
Hybrid + POS (Hist)	0.9218	0.8460	+0.0244	+2.97%
Hybrid + POS (Bigram)	0.9196	**0.8489**	**+0.0273**	**+3.32%**
Hybrid + POS (Both)	0.9200	0.8486	+0.0270	+3.29%
Hybrid + NER	0.9183	0.8456	+0.0240	+2.92%
Hybrid + POS (Both) + NER	0.9187	0.8481	+0.0265	+3.23%

Key Finding

- The Hybrid model significantly outperforms TF-IDF.
- Adding POS/NER features to the Hybrid model provides further incremental gains.
- POS Bigrams provide the highest overall improvement (+3.32%).

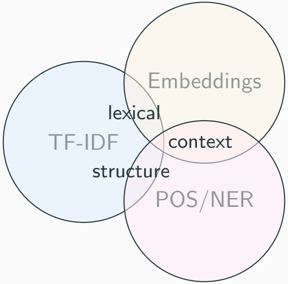
Results Visualization



Why Syntax & Entities Still Matter

Even with powerful distributional embeddings (GloVe, Doc2Vec), POS and NER features add value:

- **Non-redundant Signals:** Embeddings capture *what* words mean; POS/NER capture *how* they are used (syntax, structure, salience).
- **Error Diversity / Orthogonal Cues:** Help disambiguate semantically similar words used in different contexts (e.g., "game" in sports vs. video games).
- **Lightweight:** Computationally cheap to extract and add minimal dimensionality instead of stacking deep architectures.



Example: "game"

"The game was exciting"
↓
Sports context

"This game has graphics"
↓
Computing context

Conclusion

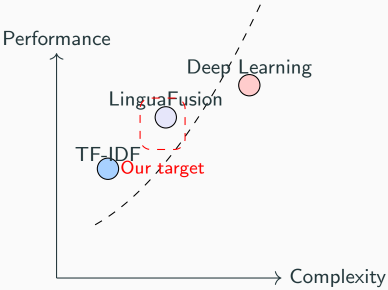
Conclusion: LinguaFusion Takeaways

Main Finding

Strategic fusion of lightweight, interpretable linguistic features (TF-IDF, POS, NER, GloVe, Doc2Vec) significantly boosts classification accuracy (+3.32 pp) over a strong TF-IDF baseline, without resorting to complex deep learning models.

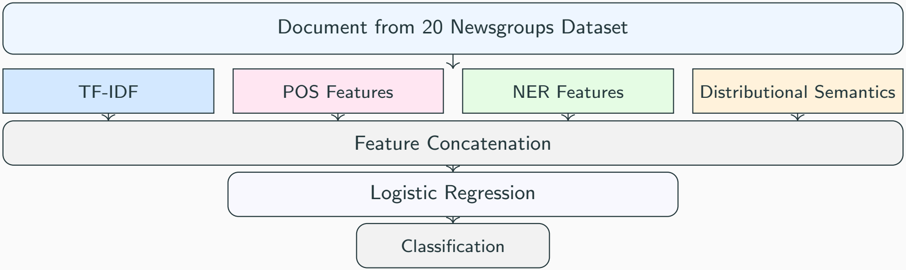
Key Insights:

- **Complementarity is Key:** Different feature types capture distinct linguistic aspects; their combination is powerful.
- **Lightweight Block:** Uses inexpensive feature blocks, enabling fast training and serving on modest hardware.
- **Interpretability is Preserved:** Each feature block has a clear linguistic meaning, aiding analysis and debugging.



LinguaFusion: Architecture Detail

Feature Block	Extractor	Dimensions	Library
TF-IDF	TfidfVectorizer	≈20,000	sklearn
POS Histogram	POS tagger	45	NLTK
POS Bigrams	POS tagger + TF-IDF	≈1,500	NLTK + sklearn
NER Histogram	Named Entity Recognition	9	spaCy
GloVe	Pre-trained embeddings	300	Stanford NLP
Doc2Vec	Trained document vectors	300	Gensim



Thank You!