Ablation Tests for ML Methods on Lab Dataset

Yi-Chun (Rimi) Chen

November 24, 2025

1 Ablation: Linear Baselines on Clause Intent Classification

1.1 Data and Input Variants

Each example is a single instance with one gold label. The JSON schema is
`<id>: {"text": ..., "span": ..., "labels": [{"label": [<y>]}]}`. We consider two input choices per instance:

- **Span**: the annotated fragment `span`.
- **Text**: the full field `text`, which typically aligns better with sentence or clause units.

Unless otherwise stated, models are trained and evaluated separately for the two choices.

Definition of document and term. In our setup, a *document* corresponds to one input example, which can vary depending on the experimental condition: (i) for sentence-level runs with **Text** input, the full sentence string is treated as the document; (ii) for sentence-level runs with **Span** input, only the annotated span within that sentence forms the document; (iii) for subsentence-level runs with **Text** input, the clause string is the document; and (iv) for subsentence-level runs with **Span** input, the annotated fragment within the clause is the document. A *term* is any feature extracted from the document, either a word n–gram (1–2) or a character n–gram (3–5). The TF–IDF weight $\mathrm{tfidf}(t, d)$ is therefore computed with respect to the current set of training documents, so idf values can differ between Text and Span runs depending on the unit of analysis.

1.2 Step 1: TF–IDF + Logistic Regression

Research question. Step 1 establishes a lexical baseline by asking: *can we classify each clause directly from its literal words and character patterns, without any semantic abstraction*? This treats intent recognition as a surface cue matching task, where performance reflects how strongly individual tokens or short n–grams correlate with labels.

Conceptual motivation. TF–IDF is a classical sparse representation that emphasizes tokens distinctive for each document. By combining word n–grams (1–2) and character n–grams (3–5), we capture both semantic content and surface form patterns, which is important in patient-provider messages where typos and short formulaic phrases are common. Logistic regression provides an interpretable linear classifier with probabilistic outputs, convex optimization, and built-in handling of class weights for imbalance. Together, TF–IDF and LR form a strong lexical memorization baseline: performance reflects direct word- and character-level evidence without semantic abstraction.

Feature extraction and loss. We build sparse bag–of–n–grams with scikit–learn. TF–IDF is defined as

$$\mathrm{idf}(t) = \log\left(\frac{N+1}{\mathrm{df}(t)+1}\right) + 1, \qquad \mathrm{tfidf}(t,d) = \mathrm{tf}(t,d) \cdot \mathrm{idf}(t).$$

The concatenated vectors x_d are classified with multinomial logistic regression using softmax, with weighted cross–entropy

$$\mathcal{L} = -\sum_{i=1}^{n} \alpha_{y_i} \log p(y_i \mid x_i),$$

where $\alpha_c \propto 1/\mathrm{freq}(c)$ addresses imbalance.

1.3 Step 2: LSA + Prototype Similarities + Logistic Regression

Research question. Whereas Step 1 evaluates classification from surface lexical evidence, Step 2 asks: *does semantic abstraction and exemplar structure improve clause classification when surface forms vary or when classes are rare*? The idea is to project clauses into a latent semantic space (LSA) to align paraphrases, and to augment them with prototype similarities to provide global class anchors.

Conceptual motivation. TF–IDF features are sparse and sensitive to wording variation. Latent Semantic Analysis (LSA) reduces sparsity by projecting into a low-rank semantic space via truncated SVD, producing dense embeddings that group related terms (e.g., *refill*, *dosage*, *medication*). We further enrich these embeddings with prototype similarities: for each class, we compute a centroid vector from its training examples and augment document vectors with their cosine similarities to these prototypes. This hybridization encodes both (a) document-level semantics and (b) closeness to class exemplars, which is expected to aid minority or ambiguous classes.

Latent semantic indexing (LSA). From the word TF–IDF matrix $X \in \mathbb{R}^{n \times m}$, truncated SVD gives

$$X \approx U_k \Sigma_k V_k^\top, \qquad k \ll m,$$

with document embeddings $z_d = (U_k \Sigma_k)_{d:}$ normalized to $\|z_d\|_2 = 1$.

Class prototypes. For each class c with training set D_c, the prototype is

$$p_c = \frac{1}{|D_c|} \sum_{i \in D_c} z_i, \quad p_c \leftarrow \frac{p_c}{\|p_c\|_2}.$$

Each document embedding z is augmented with prototype similarities $s_c(z) = z^\top p_c$, yielding $\tilde{z} \in \mathbb{R}^{k+C}$ for logistic regression.

1.4 Training Protocol and Metrics

We use an 80–20 split with stratification by label when possible. No heavy normalization is applied. Step 1 uses concatenated word and character TF–IDF; Step 2 uses only word TF–IDF with LSA and prototypes. Metrics include accuracy, macro–F1, per-class F1, and confusion matrices.

Table 1: Accuracy and Macro-F1 for filtered 7-class runs.

Step	Unit	Input	Accuracy	Macro-F1
Step 1	Sentence	Span	0.6429	0.6283
Step 1	Sentence	Text	0.5909	0.5874
Step 1	Subsentence	Span	0.6649	0.6364
Step 1	Subsentence	Text	0.5464	0.5222
Step 2	Sentence	Span	0.5779	0.5663
Step 2	Sentence	Text	0.5390	0.5254
Step 2	Subsentence	Span	0.6082	0.5861
Step 2	Subsentence	Text	0.5206	0.5029

1.5 Results

Headline comparison. Table 1 compares accuracy and macro–F1 across span and text units under both steps.

Per-class focus. Tables 2 and 3 report per-class F1 for sentence-level and subsentence-level runs, respectively.

Table 2: Per-class F1 for sentence runs (rounded to 3 decimals).

Label	S1 Sent+Span	S1 Sent+Text	S2 Sent+Span	S2 Sent+Text
Care_Coordination	0.678	0.600	0.571	0.561
Diagnostics	0.764	0.615	0.680	0.583
Drugs	0.636	0.585	0.579	0.412
General_information	0.468	0.408	0.489	0.444
Health_care_access_and_quality	0.500	0.462	0.429	0.353
Scheduling_appt	0.780	0.810	0.680	0.692
Symptoms	0.571	0.632	0.537	0.632

Table 3: Per-class F1 for subsentence runs (rounded to 3 decimals).

Label	S1 Subsent+Span	S1 Subsent+Text	S2 Subsent+Span	S2 Subsent+Text
Care_Coordination	0.781	0.615	0.730	0.576
Diagnostics	0.827	0.718	0.765	0.639
Drugs	0.656	0.586	0.567	0.607
General_information	0.491	0.351	0.444	0.327
Health_care_access_and_quality	0.424	0.389	0.410	0.286
Scheduling_appt	0.712	0.552	0.698	0.545
Symptoms	0.564	0.444	0.488	0.541

1.6 Summary of Findings

- **Lexical baselines are strong.** Step 1 achieves macro–F1 ≈ 0.6, showing that surface n–grams already capture much of the intent signal.

- **Semantic abstraction does not consistently help.** Step 2 was expected to aid paraphrases and tail classes, but performance often declined, suggesting annotation noise and imbalance outweigh latent semantics.
- **Span focus outperforms text context.** Span inputs concentrate label cues and outperform full text. However, this also shows that performance depends heavily on annotation design rather than model capacity.

Overall, results plateau around macro–F1 ≈ 0.6, indicating that further gains require either improving annotation consistency or introducing structured context beyond individual clauses.

Bridge to structured models. These ablations establish a progression of modeling assumptions—from literal n–gram matching (Step 1), to latent semantics (Step 2), to prototype-informed representations. This staged setup motivates our next experiments with graph-based methods, which extend beyond both lexical and semantic features to encode structured relations between clauses.

1.7 Span versus Text as Input Units

Span versus text runs provide complementary perspectives. Span-based results indicate the maximum potential when the model is given clean intent fragments, while text-based results approximate more realistic deployment conditions where the model must extract cues from raw sentences or clauses.

Pros of span input. Spans provide cleaner supervision and compact lexical cues, reducing noise and improving discriminability. This aligns with practices in slot filling and information extraction, where annotated spans mark the exact evidence for a label.

Cons of span input. Using spans at inference time is unrealistic, since new messages arrive without human-highlighted fragments. Moreover, our span inconsistency analysis shows that spans vary widely in length (1–49 tokens) and sometimes cut across clause boundaries. Such inconsistency weakens supervision: overly short spans yield trivial lexical matches, while overly long spans conflate multiple communicative acts. Models may thus overfit to human filtering, inflating performance.

Examples of consistent vs. inconsistent spans. Consistent span definitions are critical in prior datasets. The PDTB regulates spans as discourse arguments [6], while RST-DT enforces EDU- level segmentation [1, 2]. Similarly, ATIS [7] and SNIPS [3] define entity spans with precise token-level guidelines. By contrast, i2b2 clinical text annotations [8], stance detection in tweets [5], and event nugget corpora [4] allow variable-length fragments without strict boundaries. These heterogeneous practices mirror the variability in our dataset, where spans may be short pleasantries, single clauses, or longer cross-boundary fragments, reducing the reliability of clause-level supervision and partly explaining the plateau in our baselines.

References

[1] Carlson, L., Marcu, D.: Discourse tagging reference manual. ISI Technical Report ISI-TR-545 **54**(2001), 56 (2001)

[2] Carlson, L., Marcu, D., Okurowski, M.E.: Building a discourse-tagged corpus in the framework of rhetorical structure theory. In: Current and new directions in discourse and dialogue, pp. 85–112. Springer (2003)

[3] Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., Lavril, T., et al.: Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. arXiv preprint arXiv:1805.10190 (2018)

[4] Mitamura, T., Yamakawa, Y., Holm, S., Song, Z., Bies, A., Kulick, S., Strassel, S.: Event nugget annotation: Processes and issues. In: Proceedings of the 3rd workshop on EVENTS: Definition, detection, coreference, and representation. pp. 66–76 (2015)

[5] Mohammad, S., Kiritchenko, S., Sobhani, P., Zhu, X., Cherry, C.: Semeval-2016 task 6: Detecting stance in tweets. In: Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016). pp. 31–41 (2016)

[6] Prasad, R., Webber, B., Joshi, A.: The penn discourse treebank: An annotated corpus of discourse relations. In: Handbook of linguistic annotation, pp. 1197–1217. Springer (2017)

[7] Price, P.: Evaluation of spoken language systems: The atis domain. In: Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990 (1990)

[8] Uzuner, Ö., South, B.R., Shen, S., DuVall, S.L.: 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. Journal of the American Medical Informatics Association **18**(5), 552–556 (2011)

A Semantic Processing in NLP: From Simple to State of the Art

In this appendix, we provide background on how natural language processing (NLP) methods capture semantics, moving from simple lexical counts to modern contextual encoders. This overview clarifies where our ablation baselines (TF-IDF, LSA) fit into the broader landscape.

A.1 From Counts to Dense Representations

Bag-of-Words and TF-IDF. The simplest approach to semantics is based on lexical overlap. Bag-of-Words (BoW) and Term Frequency–Inverse Document Frequency (TF-IDF) represent text as sparse vectors indexed by tokens or n-grams. These methods rely on direct word overlap and cannot disambiguate synonyms or polysemous terms, but they provide strong baselines for tasks where surface cues correlate with labels.

Latent Semantic Analysis (LSA). LSA applies truncated singular value decomposition (SVD) to the TF-IDF matrix, projecting documents into a lower-dimensional latent space. This smooths over word choice variation by grouping tokens that co-occur in similar contexts. LSA thereby introduces a shallow form of semantic abstraction while remaining unsupervised and computationally efficient.

A.2 Static Word Embeddings

word2vec, GloVe, fastText. Static embeddings map each word type to a dense vector trained on distributional context. Similar words (e.g., "doctor," "physician") receive nearby vectors. FastText further incorporates subword information, improving handling of typos and morphology. These embeddings improved generalization but still assign one vector per word regardless of context.

A.3 Contextual Representations

ELMo and BERT-family encoders. Contextual encoders output different embeddings for the same word depending on surrounding context (e.g., "cold" as a symptom vs. weather). This ability to disambiguate polysemy and encode paraphrases marked a major advance. Models such as BERT, RoBERTa, and DeBERTa remain state of the art for classification and understanding tasks.

Sentence embeddings. Derived from contextual encoders, models such as SBERT or SimCSE produce sentence-level embeddings optimized for semantic similarity. These are widely used for clustering, intent detection, and retrieval, and would be a natural replacement for LSA in our Step 2 baseline.

A.4 Structured and Compositional Semantics

Graph-based formalisms. Structured representations such as Semantic Role Labeling (SRL), Abstract Meaning Representation (AMR), and Universal Conceptual Cognitive Annotation (UCCA) explicitly encode predicate-argument relations or discourse structure. These support compositional reasoning and align well with explainability goals. Our graph-based experiments build directly on this line of work.

A.5 Large Language Models

LLMs as semantic integrators. Instruction-tuned large language models bundle many semantic capabilities: they learn contextual embeddings, paraphrase robustness, and world knowledge. With careful prompting or light fine-tuning, LLMs can adapt to domain-specific intent tasks even with limited supervision.

A.6 How These Pieces Answer "Semantics"

- Counts approximate meaning by word overlap.
- LSA models semantic relatedness through shared usage patterns.
- Static embeddings give word meaning but not context.
- Contextual embeddings represent meaning in context, which is crucial for clause intents.
- Structured formalisms encode predicate–argument and discourse relations for compositional meaning.
- LLMs bundle many of the above, plus world knowledge, and can be adapted with minimal labeled data.