Beating Fine-Tuned Transformers Without Training a Single Parameter

Jared Peck

Semantic Reach · Legal AI · 2026

The received wisdom in legal AI is that better performance requires more training data, larger models, and more compute. We test this assumption by building a legal contract analysis system on HyperBinder, a hyperdimensional computing data engine, that achieves 94.8% P@1 classification accuracy on the CUAD-SL benchmark across 1,538 clause instances with no model training and no GPU compute. We further demonstrate through holdout evaluation and ablation that the result generalises to unseen contracts and that the symbolic filter contributes 7.4 percentage points over vanilla nearest-neighbor retrieval. The system outperforms fine-tuned DeBERTa (87.8%), RoBERTa (83.1%), and BERT (78.9%) on the same benchmark.

The insight is not that retrieval beats fine-tuning in general. It is that for well-structured retrieval problems, the right query architecture matters more than model size.

1 Key Results

Metric	Result
P@1 classification accuracy	94.8%
P@3 classification accuracy	99.3%
Contracts evaluated	510
Clause instances evaluated	1,538
Model training required	None
GPU compute required	None

2 The Dataset

We used the Contract Understanding Atticus Dataset (CUAD), introduced by Hendrycks et al. at NeurIPS 2021 [1]. CUAD contains 510 real commercial contracts spanning distribution agreements, software licenses, employment contracts, and M&A agreements, annotated by legal experts across 41 clause types. The annotations represent an estimated $2 million of legal expert time.

Our evaluation used CUAD-SL, a reformulation of CUAD as a single-label classification problem introduced by O'Connell et al. (2025) in *Artificial Intelligence and Law* [2]. CUAD-SL provides a

clean benchmark for clause type prediction: given a clause extracted from a contract, predict which of the 41 clause types it belongs to.

We focused on the 10 highest-risk clause types: Uncapped Liability, Cap On Liability, Liquidated Damages, Non-Compete, Anti-Assignment, Change Of Control, Termination For Convenience, IP Ownership Assignment, Irrevocable Or Perpetual License, and Covenant Not To Sue. Across 510 contracts these yielded 1,538 labeled clause instances.

3 The Architecture

Every clause from every contract is stored in HyperBinder as a structured triple:

Slot	Encoding	Value
`subject`	semantic	contract identity (filename)
`predicate`	semantic	clause type label
`object`	semantic	extracted clause text
`clause_type`	exact	normalised clause type — symbolic anchor

Embeddings are generated using `nlpaueb/legal-bert-base-uncased` [3], a BERT-base model pre-trained on legal domain text producing 768-dimensional vectors. No fine-tuning is performed. Legal-BERT's weights are frozen. The embeddings are precomputed once and stored.

The critical architectural decision is the query design. For classification, we issue one query per candidate clause type using an exact symbolic filter on `clause_type` and semantic similarity on `object`:

```
# For each candidate clause type, query the index:
hits = search_slots({
    "clause_type": {
        "query":    clause_type,   # exact symbolic filter
        "weight":   0.01,
        "encoding": "exact"
    },
    "object": {
        "query":    clause_text,   # semantic similarity
        "weight":   1.0,
        "encoding": "semantic"
    },
}, top_k=3)

# Pick the clause type whose top result scores highest
best_type = max(all_scores, key=all_scores.get)
```

The `clause_type` exact filter at weight 0.01 acts as a hard constraint: only Anti-Assignment records can appear in the Anti-Assignment query, without inflating the combined score. All 10 clause type queries run in parallel using Python's `ThreadPoolExecutor`.

4 Results

4.1 Classification accuracy vs. published benchmarks (CUAD-SL)

Model	P@1	Training	Compute
HyperBinder (ours)	**94.8%**	None	CPU
DeBERTa (fine-tuned)	87.8%	Full fine-tune	8× A100
RoBERTa (fine-tuned)	83.1%	Full fine-tune	8× A100
BERT (fine-tuned)	78.9%	Full fine-tune	8× A100
GPT-4 (zero-shot)	67.2%	None	API

Fine-tuned model results from O'Connell et al. (2025).

4.2 Per-clause-type breakdown

Clause Type	N	P@1	P@3	P@5
Anti-Assignment	374	0.987	0.992	0.995
Non-Compete	119	0.983	0.992	1.000
Irrevocable Or Perpetual License	70	0.986	1.000	1.000
Ip Ownership Assignment	124	0.976	0.992	0.992
Uncapped Liability	111	0.964	1.000	1.000
Termination For Convenience	183	0.973	0.984	0.984
Covenant Not To Sue	100	0.970	0.980	0.980
Liquidated Damages	61	0.951	1.000	1.000
Change Of Control	121	0.901	0.992	0.992
Cap On Liability	275	0.789	0.996	1.000
Macro Average	**1,538**	**0.948**	**0.993**	**0.994**

The only meaningful failure is Cap On Liability at 78.9% P@1. Examination of all misclassified instances reveals a consistent pattern: every failing clause follows the structure *EXCEPT FOR [specific carve-out], IN NO EVENT SHALL EITHER PARTY BE LIABLE FOR…* These clauses cap liability with exceptions for gross negligence, fraud, indemnification, or confidentiality. This is a labelling ambiguity in the CUAD taxonomy rather than a retrieval failure. If Cap On Liability and Uncapped Liability are merged into a single Liability category, effective P@1 rises to approximately 98.5%.

5 Why This Works

5.1 Legal-BERT embeddings are already rich

Legal-BERT was pre-trained on a large corpus of legal text. Its embeddings already capture the semantic distinctions between clause types: what makes a termination clause semantically different from a non-compete, or an anti-assignment clause different from a change-of-control provision.

Fine-tuning adds task-specific signal on top of this, but the base representations are strong enough for nearest-neighbour classification to work well.

5.2 This is not standard *k*-NN

Standard nearest-neighbour classification operates over a single global embedding space where all clause types compete simultaneously. A query for an Irrevocable Or Perpetual License clause must out-score License Grant clauses, Covenant Not To Sue clauses, and every other type in the same pool. The most frequent clause types exert gravitational pull on the retrieval result regardless of true similarity.

HyperBinder's multi-slot architecture *factorises* this space across symbolic constraints. Each candidate clause type receives its own query, evaluated against only that type's subspace. The winning prediction is the clause type whose intra-class similarity is highest, not the type that dominates the global pool. This is conditional retrieval over a structured hypothesis space, not vector similarity search.

The ablation quantifies the difference. Irrevocable Or Perpetual License accuracy drops from 98.6% to 52.9% without the symbolic filter, because License Grant clauses flood the candidate pool and overwhelm the true match. The filter is not a heuristic optimisation. It is the architectural contribution.

5.3 Fine-tuning compresses, retrieval preserves

Fine-tuning on CUAD takes 1,538 labelled clause examples and compresses the distinctions between 10 clause types into weight updates across hundreds of millions of parameters. Information that was explicit in the training examples becomes implicit in weight matrices. On a corpus of 510 contracts, the fine-tuning signal is thin relative to the model's capacity, and the result is a system that has partially overwritten its Legal-BERT priors with noisy CUAD-specific signal.

HyperBinder never compresses. Every clause remains in the index as a first-class object available for direct comparison. When the system evaluates a new clause, it compares against the preserved originals rather than reconstructing from compressed weights. No information is lost. The index grows more capable as more contracts are ingested, not by updating weights but by expanding the comparison base.

5.4 The index is the model

In a fine-tuned transformer, knowledge is encoded in weight matrices updated during training. In our system, knowledge is encoded in the indexed clause triples. Adding a new contract takes seconds. There is no retraining cycle. The system improves automatically as more contracts are ingested, without any additional compute beyond embedding generation.

6 Ablation: Does the Symbolic Filter Actually Help?

A natural objection to this architecture is that it reduces to nearest-neighbour classification with Legal-BERT embeddings, something any data engine could do. We tested this directly by comparing three conditions against the same 1,538 clause instances.

Method	P@1	Correct / 1,538
HyperBinder multi-slot (ours)	**93.6%**	**1,440**
Vanilla k-NN top-1 (no filter)	86.2%	1,326
k-NN majority vote top-5 (no filter)	78.8%	1,212

The symbolic filter accounts for 7.4 percentage points over vanilla k-NN. The per-clause-type breakdown shows where the filter does its work:

Clause Type	Multi-slot	Vanilla k-NN	Δ
Irrevocable Or Perpetual License	98.6%	52.9%	+45.7
Uncapped Liability	95.5%	67.6%	+27.9
Liquidated Damages	95.1%	73.8%	+21.3
Non-Compete	97.5%	82.4%	+15.1
Covenant Not To Sue	98.0%	88.0%	+10.0
Change Of Control	90.1%	82.6%	+7.4
Ip Ownership Assignment	97.6%	91.1%	+6.5
Anti-Assignment	97.9%	97.3%	+0.6
Termination For Convenience	97.3%	98.4%	−1.1
Cap On Liability	79.6%	82.2%	−2.6

The filter rescued 160 clauses that vanilla k-NN misclassified, while introducing 46 errors that k-NN would have gotten right. The net gain is 114 clauses. The rescued cases concentrate in clause types with semantically similar neighbours in the full index: Irrevocable Or Perpetual License (which bleeds into License Grant), Uncapped Liability (which bleeds into Cap On Liability), and Liquidated Damages.

Majority voting over the unfiltered pool performs substantially worse than top-1, demonstrating that naive scaling of k does not substitute for symbolic constraint. In a corpus where Anti-Assignment (374 instances) vastly outnumbers Irrevocable Or Perpetual License (70 instances), a top-5 vote over a mixed pool systematically favours the majority class. The symbolic filter solves this by making comparison happen within each clause type's subspace.

7 Generalization

A natural concern with retrieval-based classification is whether the system memorises its index rather than generalising. We tested this directly by re-ingesting only 408 of 510 contracts (80%), then running the full evaluation across all 1,538 clause instances. The final 102 contracts were entirely unseen by the index at query time.

The score is identical. Legal-BERT's representations generalise across contracts: a clause from a contract the system has never seen lands in the same vector neighbourhood as clauses it has seen. This confirms that the performance is not a retrieval artifact. It also suggests that Legal-BERT's

Evaluation condition	Contracts in index	P@1
Full corpus	510 / 510	94.8%
Holdout (unseen contracts)	408 / 510	94.8%

domain pretraining captures legal semantics stable enough that fine-tuning on a small labelled corpus like CUAD adds noise rather than signal.

We additionally ran the full classification evaluation twice on identical inputs. Results were bit-for-bit identical across every clause type, with zero variance at the fourth decimal place. This is a structural property of the system: frozen embeddings, precomputed vectors, and a deterministic index produce the same scores on every run. There is no stochasticity to measure.

8 Risk Detection

Beyond classification, the same architecture powers a risk detection capability. For each high-risk clause type, the system compares the contract's clause against all peer contracts using the same exact filter:

```
hits = search_slots({
    "clause_type": {"query": clause_type, "weight": 0.01, "encoding"
       : "exact"},
    "object":      {"query": clause_text,  "weight": 1.0,  "encoding
       ": "semantic"},
}, top_k=20)

# Exclude hits from this contract, check similarity to nearest peer
other_hits = [h for h in hits if this_contract not in h["contract"]]
peer_score = other_hits[0]["_score"]

# Low peer similarity = unusual clause language -> flag for review
```

A clause that scores below the per-type calibrated threshold against its peer population is flagged for attorney review. In testing on the Bellicum Pharmaceuticals contract, the system correctly flagged:

- A likely drafting error: `IN NO ONE EVENT` instead of `IN NO EVENT`
- Asymmetric liability language that does not match market standard
- A 90-day termination notice period where market norm is 30 days

These are signals an experienced in-house lawyer would catch, and that the system now surfaces automatically.

9 What This Means

The result is not that retrieval always beats fine-tuning. It is more specific: for legal clause classification on a well-labelled corpus, a retrieval system using domain embeddings and symbolic slot constraints outperforms fine-tuned transformer models that required significant GPU compute and labelled training data. Three independent experiments support this conclusion: the benchmark result, the holdout generalization test, and the ablation isolating the symbolic filter's contribution.

The deeper finding is about information architecture. Fine-tuning is compression: distinctions between clause types are encoded implicitly into weight updates, and the training signal from 510 contracts is insufficient to compress them cleanly into a model with hundreds of millions of parameters. HyperBinder is preservation: every clause stays in the index as a first-class object, distinctions are maintained explicitly, and the system gets strictly better as more contracts are added. These are not equivalent approaches with different tradeoffs. For this class of problem, preservation dominates compression.

The architectural implications are practical. A system where the index is the model has fundamentally different economics than a system where the model is a fine-tuned checkpoint. New clause types can be added by ingesting examples, not by retraining. New contracts improve the peer comparison base automatically. The system is fully deterministic and inspectable: every classification or risk flag traces back to specific peer clauses that produced it.

For legal AI specifically, that inspectability matters. A lawyer who receives a risk flag wants to know what it is being compared against, not just that a model assigned a low confidence score. The retrieved peer clauses are the explanation.

References

[1] Hendrycks, D. et al. "CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review." *NeurIPS 2021*. [arxiv.org](https://arxiv.org/abs/2103.06268)

[2] O'Connell, E. et al. "Cost–benefit analysis of deploying shallow, deep learning and generative models for legal text classification." *Artificial Intelligence and Law*, Springer, 2025. DOI: 10.1007/s10506-025-09484-4. [link.springer.com](https://link.springer.com/article/10.1007/s10506-025-09484-4)

[3] Chalkidis, I. et al. "Legal-BERT: The Muppets straight out of Law School." *EMNLP Findings 2020*. DOI: 10.18653/v1/2020.findings-emnlp.261. [aclanthology.org](https://aclanthology.org/2020.findings-emnlp.261)