Computer Life
ISSN: 1819-4818 | Vol. 15, No. 1, 2026

Retrieval-Augmented Legal Judgment Prediction with Adaptive Case Retrieval

Anand Hayes, Emma Lewis

School of Informatics, Computing, and Engineering, Indiana University Bloomington, USA

Abstract: Legal judgment prediction (LJP) models are mostly parametric, although judges routinely consult precedents, and recent retrieval-augmented LJP systems attach a fixed number of precedents to every case. We propose adaptive case retrieval (ACR), which treats a calibrated parametric predictor and the precedent votes of a sparse and a dense retriever at seven retrieval depths as experts, and learns a gate that mixes them from confidence, similarity, label-dispersion and agreement features. The dense retriever is a judgment-consistent contrastive (JCC) encoder whose objective adapts the contrastive knowledge-infusion loss of LegalReasoner by treating cases with agreeing judgments as positives. On CAIL2018-Small with train–test duplicates removed (112 charges, 91 law articles, 11 term intervals), JCC raises 10-nearest-neighbour charge accuracy from 76.9% to 90.7%, and ACR improves a TF-IDF+SVM predictor by 0.62 and 1.28 accuracy points on law-article and prison-term prediction ($p < 0.001$), matching the best validation-tuned fixed-k interpolation in distribution. Its advantage grows when the precedent memory expands without re-training: with every component fitted on 20% of the labels, expanding the memory to the full training set raises term accuracy by 1.87 points under ACR versus 0.53 points under the fixed scheme, recovering 86.8% of the gap to a fully re-trained model, and the gains do not stem from near-duplicate precedents. A confidence-based routing rule bypasses retrieval for the 70–90% most confident cases while preserving the full charge and article gains, and the complete pipeline costs about 11 ms per case on one CPU core.

Keywords: Legal judgment prediction; Retrieval augmentation; Adaptive retrieval; Legal case retrieval; Contrastive learning; Mixture of experts.

1. Introduction

Legal judgment prediction (LJP) infers the outcome of a case, typically the applicable law articles, the charges and the term of penalty, from the textual description of its facts [1], [2]. Besides serving as a demanding benchmark for legal language understanding, LJP supports practical tasks such as drafting assistance, consistency checking and caseload triage, and it has been studied for legal systems as different as the European Court of Human Rights [3], [4] and Chinese criminal courts [5], [6].

Judges rarely decide a case in isolation. They compare it with similar decided cases in order to keep charging and sentencing consistent, a practice that also drives research on legal case retrieval [7], [8]. Most LJP models are nonetheless purely parametric: topological multi-task learners [6], models that separate confusing law articles [9] and circumstance-aware architectures [10] compress the entire training corpus into network weights. Recent studies have begun to inject precedents explicitly. PLJP lets domain models propose candidate labels and precedents that a large language model (LLM) then reads in context [11]; CR4LJP attaches the representations of the top-k most similar historical cases to the fact encoder [12]; and LegalReasoner [13] places a graph-based case-law retrieval stage between LLM knowledge infusion and multi-hop reasoning. Its ablation study is instructive: removing the retrieval stage lowers accuracy by 1.6 points on ECHR and by 1.2 points on CAIL2018, which confirms that precedents carry information that the parametric components do not fully capture.

These systems share one design decision that we believe is suboptimal: every case receives the same number of precedents, and the retrieved evidence is trusted to the same degree no matter how reliable it is. Case bases are heterogeneous. A routine theft case usually has dozens of near-identical precedents whose verdicts agree, so a deep vote over many neighbours is safe. A case on the boundary between two confusable charges, or one involving a rare offence, may instead have neighbours that are lexically close but legally different; there the parametric model, or at most the one or two closest precedents, should dominate. A fixed retrieval depth cannot express this trade-off. Retrieving for every case also spends computation on cases that the parametric model already decides with high confidence; avoiding that cost requires a routing decision taken before retrieval, not merely a different weighting of the retrieved evidence. Adaptive retrieval has been explored for open-domain question answering, where systems decide when to retrieve on the basis of entity popularity [14], self-reflection [15] or query complexity [16], and adaptive neighbour selection improves nearest-neighbour machine translation [17]. To our knowledge, these ideas have not been examined for judgment prediction, where the retrieved items are labelled decisions rather than passages of text.

We study retrieval-augmented LJP with adaptive case retrieval (ACR). ACR treats a parametric predictor and a family of precedent-voting distributions, obtained from two complementary retrievers at depths $k \in \{1, 2, 4, \ldots, 64\}$, as experts, and learns a light per-case gate that decides how much each expert contributes. The gate reads signals that a judge would also weigh: how confident the parametric model is, how similar the closest precedents are, how strongly the verdicts of the top-k precedents agree, and whether lexical and semantic evidence point to the same outcome. The second retriever is a judgment-consistent contrastive (JCC) encoder. We adapt the contrastive knowledge-infusion objective of LegalReasoner so that two cases count as positives when their judgments agree, which yields a similarity measure aligned

with legal outcomes rather than with surface wording.

We evaluate ACR on CAIL2018-Small under a leakage-controlled protocol in which every fusion parameter is fitted on validation data only. Our contributions are threefold.

1) We cast the use of precedents in LJP as gated mixing over retrievers and retrieval depths, and train the gate by maximum mixture likelihood, whose gradient has a direct credit-assignment interpretation.

2) We adapt the contrastive knowledge-infusion objective of LegalReasoner into a judgment-consistent dense case retriever that is trained without a deep-learning framework and lifts 10-NN charge accuracy on validation cases from 76.9% to 90.7%.

3) Through significance tests, ablations and targeted analyses we characterise where precedents help: they improve article and term prediction significantly, matter most for sentencing, change decisions mainly for low-confidence cases, which a confidence-based routing rule exploits to bypass retrieval, and, for prison terms under ACR, keep paying off as new precedents are added without re-training.

2. Related Work

2.1. Legal Judgment Prediction

Early LJP systems paired n-gram and topic features with linear classifiers; Aletras et al. [3] predicted decisions of the European Court of Human Rights with a linear SVM. Neural models followed. Luo et al. [5] predicted charges jointly with the supporting articles, Hu et al. [18] annotated discriminative legal attributes to separate few-shot and confusing charges, TopJudge [6] modelled the dependencies among subtasks as a directed acyclic graph, MPBFN [19] added forward prediction and backward verification across subtasks, LADAN [9] distilled distinguishing features of confusable law articles with a graph operator, and NeurJudge [10] split the fact description into circumstances guided by intermediate subtask results. Pre-trained encoders such as LEGAL-BERT [20] and Lawformer [21] and, more recently, LLM-centred frameworks [11] have pushed accuracy further. All of these improve the parametric model itself. ACR is complementary: it wraps any predictor that outputs class probabilities, so stronger backbones can be plugged in without changing the fusion procedure.

2.2. Precedents and Legal Case Retrieval

Legal case retrieval aims to find prior cases relevant to a query case. LeCaRD [7] provides expert relevance judgments for Chinese criminal cases, and BERT-PLI [8] models paragraph-level interactions between long case documents. For judgment prediction, CR4LJP [12] retrieves the top-k historical cases and fuses their representations, LegalReasoner retrieves a fixed number K of precedents and statutes by cosine similarity over graph-neural-network node representations, and PLJP [11] retrieves precedents that an LLM reads in context. In each case the amount of retrieved evidence is fixed in advance. Our dense retriever is trained with judgment supervision, in the spirit of supervised dense retrieval [22] and supervised contrastive learning [23]. Neural-symbolic dual-indexing demonstrates the value of organizing semantic and structured evidence through complementary retrieval pathways in large knowledge collections [31].

2.3. Retrieval Augmentation and Adaptive Retrieval

Retrieval-augmented generation conditions a generator on retrieved passages [24], whereas nearest-neighbour language models [25] and nearest-neighbour machine translation [26] interpolate the model distribution with a distribution read off the labels of retrieved neighbours, using a fixed number of neighbours and a fixed interpolation weight. Adaptive kNN-MT [17] replaces the fixed choice with a Meta-k network that weighs several neighbour counts using retrieval distances and the number of distinct target tokens among the neighbours. For LLMs, adaptive retrieval decides whether to retrieve at all [14]–[16]. ACR brings these ideas to LJP and makes a joint decision on three questions, namely whether to trust precedents, which retriever to trust and how deep to look, using both parametric-confidence and precedent-consistency signals. We include the Meta-k formulation as a baseline.

3. Methodology

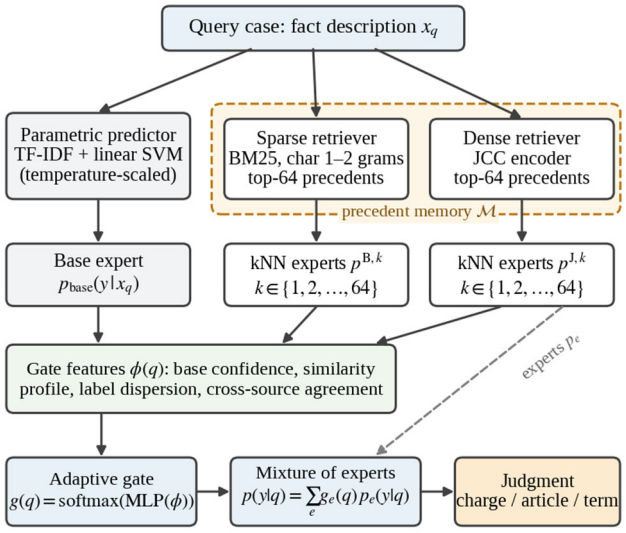


Fig. 1 Overview of adaptive case retrieval (ACR). A parametric predictor and kNN experts built from two retrievers at seven depths are mixed per case by a gate that reads confidence, similarity, dispersion and agreement features.

3.1. Problem Formulation

Let x be the fact description of a criminal case. LJP predicts a label y^t for each subtask $t \in$ {charge, article, term}, where the term of penalty is discretised into intervals. The labelled training cases $M = \{(x_j, y_j)\}$ play two roles: they train the parametric components and they form the *precedent memory* that is searched at inference time. A retriever r maps a query case q to a ranked list of precedents $j_1, \ldots, j_{64}$ with scores $s_1 \geq \ldots \geq s_{64}$. The three subtasks share the retrievers but each has its own gate, because the useful amount of evidence differs between them (Section V-C). Fig. 1 summarises the framework.

3.2. Parametric Base Predictor

The base expert is a one-vs-rest linear support vector machine over TF-IDF weighted character uni- and bi-grams, which avoids word segmentation and captures most two-character Chinese words. Margins $f(q)$ are converted into a distribution by temperature scaling [27]:

$$p_{base}(y \mid q) = softmax\left(\frac{f(q)}{T}\right) \quad (1)$$

where T minimises the negative log-likelihood on the validation split. Calibration matters here because the gate compares the base distribution with neighbour votes and uses its confidence as an input feature.

3.3. Precedent Retrievers

1) Sparse retriever: BM25 [28] ranks precedents by weighted lexical overlap over character uni- and bi-grams (k_1 = 1.2, b = 0.75), ignoring n-grams that occur in more than 20% of the cases. Because raw BM25 scores grow with query length, we use the relative score $\hat{s}_j = s_j/s_1$ when weighting neighbours.

2) Judgment-consistent contrastive (JCC) retriever: Lexical overlap is a weak proxy for legal similarity; two thefts may share few characters, whereas a theft and a robbery can be described almost identically. We therefore learn a dense encoder $h(x) = W^\top x/\|W^\top x\|$ that maps the TF-IDF vector x to a 128-dimensional unit vector. W is initialised with latent semantic analysis [29] and trained on batches that contain four cases of every charge. For an anchor case i and another case a in the batch let

$$s_{ia} = \frac{h_i^\top h_a}{\tau} \quad (2)$$

with temperature τ = 0.1. The loss of anchor i is a soft-target InfoNCE objective

$$L_i = -\sum_{a \neq i} q_{ia} \, log \frac{exp(s_{ia})}{\sum_{b \neq i} exp(s_{ib})} \quad (3)$$

whose target distribution is defined by judgment agreement,

$$q_{ia} \propto I[c_a = c_i \wedge l_a = l_i](1 + \eta \, I[t_a = t_i]) \quad (4)$$

where c, l and t denote charge, law article and term interval, I[·] is the indicator function and η = 1. Cases with the same charge and article are positives, and positives that also share the term interval receive twice the target mass, so the space reflects sentencing severity as well as the offence. The objective adapts the contrastive knowledge-infusion loss of LegalReasoner [13], which takes the InfoNCE form: with η = 0 and a single positive per anchor, (3) reduces to it, and with several judgment-defined positives it becomes a soft-target variant in the spirit of supervised contrastive learning [23]. The adaptation lies in the source of positives: LegalReasoner contrasts paraphrases or related legal concepts to infuse domain knowledge into an LLM, whereas we draw positives from recorded judgments so that the encoder itself becomes a precedent retriever. Gradients of (3) are computed in closed form and applied with row-sparse Adam updates, so training needs no deep-learning framework.

3.4. Precedent Experts

For retriever r and depth k from the depth set K = {1, 2, 4, 8, 16, 32, 64}, the precedent expert is a similarity-weighted vote of the top-k precedents,

$$p^{r,k}(y \mid q) = (1 - \varepsilon) \sum_{j=1}^{k} w_j^{r,k} I[y_j = y] + \frac{\varepsilon}{C} \quad (5)$$

$$w_j^{r,k} = \frac{exp\left(\frac{\hat{s}_j}{\tau_r}\right)}{\sum_{j'=1}^{k} exp\left(\frac{\hat{s}_{j'}}{\tau_r}\right)} \quad (6)$$

where C is the number of classes, $\varepsilon = 10^{-3}$ keeps every class possible and τ_r is tuned on the validation split. Small k trusts the closest precedent; large k approaches a local class prior around the query. With two retrievers the pool E contains 1 + 2 × 7 = 15 experts.

3.5. Adaptive Case Retrieval Gate

ACR predicts with a per-case mixture over the expert pool,

$$p(y \mid q) = g_0 p_{base} + \sum_{r,k} g_{r,k} p^{r,k} \quad (7)$$

where the weights $g = g(q)$ depend on the case and every expert is evaluated at $(y \mid q)$. The weights are produced by a one-hidden-layer gate,

$$g(q) = softmax(W_2 \, tanh(W_1 \varphi(q) + b_1) + b_2) \quad (8)$$

The feature vector $\varphi(q)$ has about 80 standardised entries in four groups. (i) *Parametric confidence*: the maximum probability, top-two margin and normalised entropy of p_{base}. (ii) *Similarity profile*: the top-1 score of each retriever (for BM25 also its length-normalised value) and the normalised scores at ranks 2, 4, 8, 16, 32 and 64. (iii) *Label dispersion*: for every expert, the vote share of its leading label and the number of distinct labels among the top-k precedents divided by k. (iv) *Agreement*: whether each expert's leading label equals the base prediction, the probability each expert assigns to that prediction, and whether the BM25 and JCC top-8 votes agree. The logarithm of the fact length completes the vector. Setting $g_0 \approx 1$ discards the retrieved evidence, so the gate decides *whether* to rely on precedents, *which* retriever to trust and *how many* neighbours to read. Because $\varphi(q)$ is computed from the retrieved lists, both searches are executed for every case; the gate changes how the evidence is weighted, not how much of it is retrieved.

The gate is trained on a held-out labelled set V by minimising the negative log-likelihood of the mixture,

$$L_{gate} = -\frac{1}{|V|} \sum_{(q,y*)} log(\sum_e g_e(q) p_e(y* \mid q)) \quad (9)$$

where the outer sum runs over the cases $(q, y^*) \in V$ with gold label y^* and the inner sum over the experts $e \in E$, plus an L2 penalty $\lambda(\|W_1\|^2 + \|W_2\|^2)$. For one case, the gradient with respect to the gate logit z_e is

$$\frac{\partial L_{gate}}{\partial z_e} = g_e(q)\left(1 - \frac{p_e(y* \mid q)}{p(y* \mid q)}\right) \quad (10)$$

which has a simple credit-assignment reading: an expert gains weight exactly when it gives the correct judgment more probability than the current mixture does. We optimise the objective with L-BFGS using this analytic gradient. The gate is fitted on the validation split rather than on the training cases, because the base model is over-confident on cases it was trained on and the training precedents would retrieve themselves; Adaptive kNN-MT likewise trains its Meta-k network on development data [17].

3.6. Inference Cost

At inference ACR computes one base prediction, two top-64 searches, 14 weighted votes and one small network evaluation. Experts and features cost O(64) per case and the gate costs O($dh + h|E|$), so the cost is dominated by the two searches, which ACR runs for every case. Adaptive fusion changes the weighting of retrieved evidence; computational savings occur only when confidence-based routing bypasses retrieval. Such a rule can only use information available before any search, namely the confidence of the base predictor; Section V-F evaluates this option.

4. Experimental Setup

4.1. Dataset

We use CAIL2018-Small, the exercise-stage release of the Chinese AI and Law Challenge [2], which contains 154,592,

17,131 and 32,508 criminal cases in its official training, validation and test files. Following common practice [6], [9], we keep cases with a single defendant, a single charge and a single law article, discard facts shorter than ten characters, and keep charges and articles that have at least 100 training cases, applying the two frequency filters until both hold. Retrieval adds a requirement that classification benchmarks usually ignore: a test case must not find itself in the memory. We therefore remove duplicated facts from the training set and drop the 371 validation and 844 test cases whose normalised fact text also appears in training. Prison terms are mapped to the eleven intervals of [6]: death penalty, life imprisonment, more than ten years, 7–10, 5–7, 3–5, 2–3 and 1–2 years, 9–12 and 6–9 months, and at most six months. Table I summarises the resulting splits.

TABLE I. Statistics of the Preprocessed CAIL2018-Small Splits

Statistic	Value
Training / validation / test cases	94,938 / 11,727 / 21,885
Charges / law articles / term intervals	112 / 91 / 11
Average fact length (characters)	407 / 400 / 413
Training cases per charge (min–max)	100–1,901
Removed val. / test cases seen in training	371 / 844
Largest term interval (≤ 6 months)	31.6% of training cases

4.2. Baselines

All systems share the same base predictor, retrievers and precedent memory, so differences isolate how precedents are used. (1) *Base*: the temperature-scaled SVM alone. (2) *kNN vote*: precedent voting with BM25 or JCC, with the depth chosen on validation. (3) *Fixed-k interpolation*: $p = (1 - \lambda)p_{\text{base}} + \lambda p^{\text{r,k}}$, the scheme of *k*NN-LM [25] and *k*NN-MT [26], with retriever, depth and λ grid-searched on validation. It corresponds to the fixed top-K precedent use of LegalReasoner's case-law retrieval stage [13] and of CR4LJP [12], transplanted to our retrievers. The selected settings are JCC, $k = 8$, $\lambda = 0.3$ for charges; JCC, $k = 32$, $\lambda = 0.4$ for articles; and BM25, $k = 8$, $\lambda = 0.4$ for terms. (4) *Confidence-triggered retrieval*: fixed-*k* interpolation applied only when the base confidence is below a threshold tuned on validation, following the retrieve-only-when-needed idea of [14]. (5) *Meta-k gate*: the Meta-*k* network of Adaptive *k*NN-MT [17] transplanted to LJP; it mixes the base expert and the seven BM25 experts using only retrieval scores and distinct-label counts, as in the original design. Because ACR only needs class probabilities, neural LJP models such as TopJudge, LADAN or LLM-based pipelines can replace the base predictor directly; the experiments use a single-CPU budget.

4.3. Implementation Details

All components are implemented in Python with NumPy, SciPy and scikit-learn and run on one CPU core with 4 GB of memory. The TF-IDF vocabulary keeps the 200,000 most frequent character uni- and bi-grams (minimum document frequency 3, sublinear term frequency) of the first 1,500 characters of each fact. The SVM regularisation constant is chosen from {0.1, 0.3, 1} by validation accuracy, giving C = 1, 1 and 0.3 for charge, article and term. The JCC encoder uses $d = 128$, $\tau = 0.1$, $\eta = 1$, a learning rate of 10^{-3}, 2000 steps with batches of 448 cases, and the checkpoint with the best 10-NN charge accuracy on 3,000 validation cases. On those cases 10-NN accuracy rises from 76.90% for the LSA initialisation to 90.70% after training. Each retriever returns 64 precedents; τ_r is chosen from {0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1}. Gate hyper-parameters (hidden size 16, 32 or 64; $\lambda \in \{10^{-3}, 10^{-2}\}$) are selected by five-fold cross-validation on the validation split, after which the gate is refitted on the full validation split. Every fusion parameter of every system is fitted on validation data only, and the test split is used once.

4.4. Evaluation Metrics

Following the LJP literature [6], [9], we report accuracy (Acc) and macro-averaged F1 for each subtask. Macro-averaged precision and recall were also computed; they follow the same trends and are omitted for space. Macro scores weight every charge, article and term interval equally and therefore expose behaviour on rare classes. Differences between systems are tested with 1,000 paired bootstrap resamples of the test set and with McNemar's test [30].

5. Results and Analysis

5.1. Main Results

Table II reports test results. Three observations stand out. First, precedents alone are informative only when the retriever is aligned with judgments. The JCC vote reaches 89.14% charge accuracy and 89.33% article accuracy, close to or above the SVM (89.39% and 88.80%), whereas the BM25 vote trails the SVM by 7.5 and 9.2 points. For prison terms both votes fall below the base model (41.36% and 40.44% versus 42.70%): neighbours share the offence far more often than the sentencing circumstances that determine the term.

Second, combining the parametric model with precedents helps on every subtask. ACR raises accuracy over the base by +0.14, +0.62 and +1.28 points for charges, articles and terms (Table III). The article and term gains are significant under both the paired bootstrap and McNemar's test ($p < 10^{-4}$ and $p < 10^{-4}$), while the charge gain is not (bootstrap CI [-0.05, 0.35]). Macro-F1 moves in the same direction (+0.05, +0.33 and +0.92), but its confidence intervals include zero because rare classes contribute few test cases.

Third, the per-case gate matches the accuracy of the best static fusion. Against fixed-*k* interpolation, whose retriever, depth and weight were grid-searched on the same validation data, ACR differs by −0.09, +0.13 and +0.16 accuracy points; only the article gain is significant (bootstrap CI [0.03, 0.23], McNemar $p = 0.013$). On terms, the macro-F1 values of the two schemes are statistically indistinguishable (32.41% versus 32.98%). ACR is clearly better than the Meta-*k* gate on articles (+0.57 points, $p < 10^{-4}$) because Meta-*k* only mixes BM25 experts, which are weak for charges and articles; on terms, where BM25 is the useful retriever, the two gates are

on par (+0.01). Gate training is stable: across three random initialisations accuracy varies by at most 0.08 points.

The confidence-triggered baseline reveals where retrieval matters. With the threshold tuned on validation, it retrieves for only 11.4% of the charge cases and 11.3% of the article cases, yet it matches always-on fixed-*k* interpolation (89.62% and 89.28%). For terms the tuned threshold routes 70.6% of the cases to retrieval. Finally, an oracle that picks the best of the 15 experts for every case would reach 94.34%, 94.08% and 72.97%, so the expert pool contains far more complementary evidence than any of the gates recovers.

TABLE II. Test Results on CAIL2018-Small (%)

Method	Charge		Article		Term	
	Acc	F1	Acc	F1	Acc	F1
TF-IDF + linear SVM (base)	89.39	87.54	88.80	86.53	42.70	31.49
kNN vote, BM25	80.17	76.25	81.27	75.63	41.36	29.92
kNN vote, JCC	89.14	87.24	89.33	86.73	40.44	27.48
Fixed-k interpolation [25]	89.62	87.65	89.28	86.89	43.82	32.98
Confidence-triggered retrieval	89.62	87.65	89.28	86.88	43.80	32.96
Meta-k gate [17]	89.40	87.49	88.84	86.47	43.97	32.99
ACR (ours)	89.53	87.59	89.41	86.86	43.98	32.41

Acc = accuracy, F1 = macro-F1; best value per column in bold. All fusion parameters are selected on the validation split.

TABLE III. Paired Significance Tests of ACR on the Test Split

Comparison	Task	ΔAcc (pp) [95% CI]	ΔF1 (pp) [95% CI]	McNemar p
vs. base	Charge	+0.14 [-0.05, 0.35]	+0.05 [-0.32, 0.38]	0.1775
	Article	+0.62 [0.40, 0.85]	+0.33 [-0.18, 0.78]	<10−4
	Term	+1.28 [0.74, 1.82]	+0.92 [-0.42, 2.28]	<10−4
vs. fixed-k	Charge	−0.09 [-0.20, 0.02]	−0.06 [-0.23, 0.13]	0.1108
	Article	+0.13 [0.03, 0.23]	−0.03 [-0.24, 0.14]	0.0129
	Term	+0.16 [-0.32, 0.65]	−0.57 [-1.72, 0.44]	0.5001
vs. Meta-k	Charge	+0.13 [-0.06, 0.34]	+0.11 [-0.27, 0.44]	0.2254
	Article	+0.57 [0.37, 0.80]	+0.38 [-0.10, 0.84]	<10−4
	Term	+0.01 [-0.42, 0.43]	−0.58 [-1.66, 0.36]	0.9642

Δ = ACR minus comparison; CIs from 1,000 paired bootstrap resamples.

5.2. Ablation Study

Table IV removes one component at a time while keeping the gate hyper-parameters of the full model. Differences below about 0.2 points lie within the paired bootstrap noise on this test set, so we focus on larger effects. The two retrievers matter for different subtasks. Without the JCC retriever, article accuracy drops by 0.58 points; without BM25, term accuracy drops by 0.75 points and term macro-F1 by 2.62 points. Replacing JCC with its LSA initialisation costs 0.57 article points, which isolates the effect of judgment-consistent training. Term-aware weighting of positives has little effect: the encoder trained with $\eta = 0$ performs comparably or slightly better (89.65% on charges and 44.19% on terms), indicating that sentencing cues are captured mainly by lexical matching of amounts and circumstances, which BM25 contributes to the mixture. Removing the parametric-confidence or agreement features changes accuracy by at most a few hundredths of a point, which indicates that the vote-share and dispersion features already carry the information the gate uses. Adaptive depth matters for sentencing: restricting each retriever to its single best depth leaves charges and articles unchanged but lowers term accuracy by 0.33 points and term macro-F1 by 0.87 points.

TABLE IV. Ablation Study on the Test Split (Acc / Macro-F1, %)

Variant	Charge		Article		Term	
	Acc	F1	Acc	F1	Acc	F1
ACR (full model)	89.53	87.59	89.41	86.86	43.98	32.41
w/o JCC retriever (BM25 only)	89.41	87.53	88.83	86.43	43.96	32.68
w/o BM25 retriever (JCC only)	89.52	87.61	89.36	86.96	43.24	29.79
JCC replaced by LSA	89.37	87.48	88.84	86.51	44.18	32.42
JCC with $\eta = 0$	89.65	87.92	89.30	87.18	44.19	32.56
w/o confidence features	89.52	87.54	89.43	86.87	44.04	32.49
w/o agreement features	89.53	87.55	89.42	86.85	43.97	32.42
Fixed depth, adaptive weight	89.57	87.63	89.40	86.92	43.66	31.54

5.3. What Does the Gate Learn?

Fig. 2 shows how the trained gates distribute their weight. The parametric expert keeps 0.57 of the mass for charges, 0.47 for articles and only 0.17 for terms. For charges and articles almost all precedent weight goes to JCC (0.37 and 0.47, against 0.06 and 0.06 for BM25), concentrated on the

deepest vote. For terms, BM25 receives 0.48, spread over depths 2, 8 and 64, and JCC receives 0.36. The gate thus recovers from validation data alone the division of labour found by the ablation, with judgment-aligned evidence for the offence and lexical evidence for sentencing. Per-case variation, however, is small. Across quintiles of base confidence the average parametric weight varies by at most 0.004 for charges and articles and stays between 0.14 and 0.19 for terms. In this dataset the gate therefore behaves mostly as a learned task-level allocation, which explains why it performs like the best static interpolation and why the confidence features contribute little.

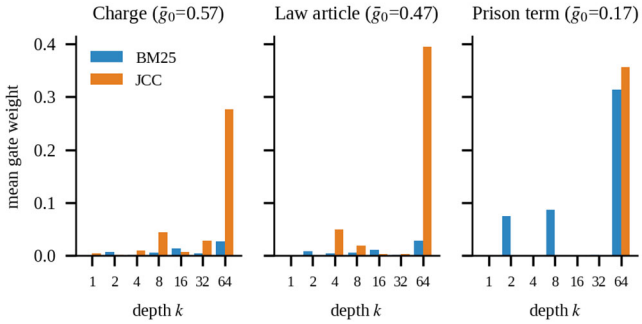


Fig. 2. Mean gate weight per retriever and depth on the test split; ḡ0 is the mean weight of the parametric expert.

5.4. Adding Precedents Without Re-training

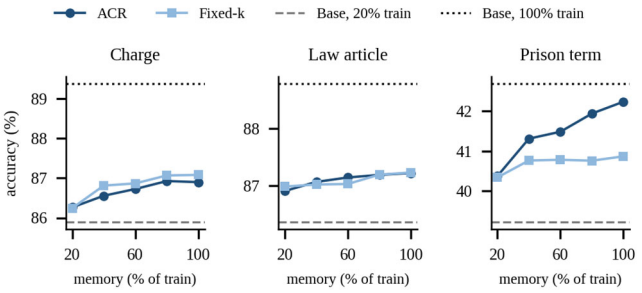


Fig. 3 Test accuracy when the base SVM, the JCC encoder, the gate and the fixed-k weights are fitted with 20% of the training labels and the precedent memory grows to 100% without re-training.

Courts publish new judgments continuously, so a practical system should improve when precedents are added, ideally without re-training. To test this, we fit every learned component, namely the SVM, the JCC encoder, the gate and the fixed-k configuration, on a 20% subset of the training labels (18,987 cases) with a memory of the same 20%, and then let the memory grow to 40–100% of the training cases. Only TF-IDF and BM25 statistics, which need no labels, are shared with the full setting. Fig. 3 shows the result. For charges and articles both fusions improve steadily and end close together (ACR 86.90% versus 87.08% for charges, 87.22% versus 87.23% for articles). For terms the curves separate. ACR gains 1.87 accuracy points as the memory grows, against 0.53 for fixed-k interpolation, and at full memory it leads by 1.37 accuracy and 3.42 macro-F1 points. It reaches 42.24%, which recovers 86.8% of the gap between the 20% SVM (39.21%) and an SVM re-trained on all labels (42.70%). The configuration selected with the small memory relies on a single deep JCC vote, whose term accuracy barely moves as cases are added. ACR spreads its weight over both retrievers and several depths; we attribute its steady gain to the BM25 neighbours, which become lexically closer, and hence more informative about the sentence, as the memory grows. The parametric weight stays at 0.09 for every memory size, so the gain comes from sharper precedent votes rather than from shifting weight away from the parametric expert. Self-play contrastive fine-tuning for web search agents highlights the importance of assessing retrieval robustness when information sources and query conditions change [32].

5.5. Rare Classes and Near-Duplicate Precedents

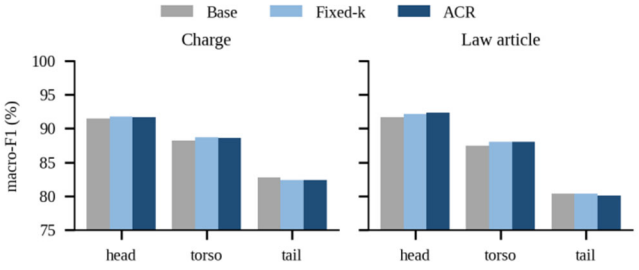


Fig. 4 Macro-F1 over classes grouped into tertiles by training frequency (head, torso, tail).

Retrieval augmentation is often motivated by the long tail [14]. Fig. 4 groups classes into tertiles by training frequency; the tail holds charges with 100–323 training cases. ACR improves macro-F1 on head and torso classes (charges 91.50 → 91.70 and 88.23 → 88.63; articles 91.67 → 92.31 and 87.44 → 88.01) but not on tail classes (82.79 → 82.33 and 80.32 → 80.07). The precedent memory is imbalanced in the same way as the training data, so the votes for a rare charge are easily outnumbered by neighbours from a frequent, similar charge. Class-balanced voting or per-class memory quotas are natural remedies.

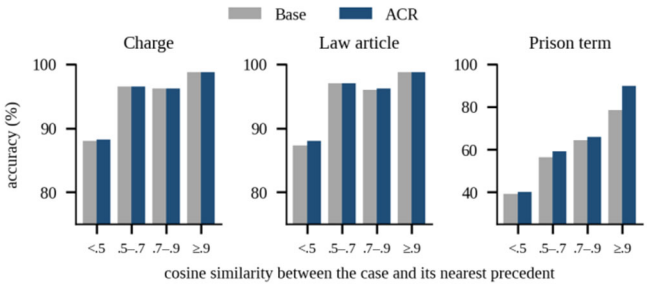


Fig. 5 Test accuracy grouped by the TF-IDF cosine similarity between each case and its nearest BM25 precedent.

A retrieval gain could also be inflated by near-copies of test cases in the memory. Exact duplicates were removed (Section IV-A), and Fig. 5 groups test cases by the cosine similarity between each case and its nearest precedent. Most cases have no close neighbour: 84.4% fall below 0.5 (median 0.28) and only 487 (2.2%) exceed 0.9. For charges and articles almost the entire gain comes from the low-similarity group (88.03% → 88.20% and 87.28% → 88.00%). For terms the gain grows with similarity, reaching 11.3 points above 0.9, because almost identical facts tend to receive the same sentence. This group contributes 19.6% of the overall term gain; without it the gain is still 1.06 points.

5.6. Confidence-Based Routing and Efficiency

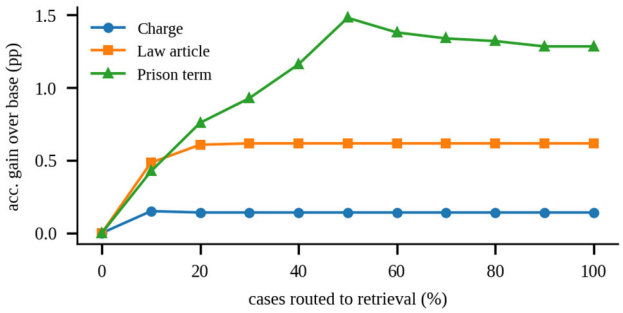


Fig. 6. Accuracy gain over the base model when only the least confident fraction of cases is routed to ACR; the remaining cases keep the base prediction and bypass both retrievers.

TABLE V. Per-Case Inference Cost of the Full Pipeline Without Routing (One CPU Core, Charge Subtask)

Component	ms / case
TF-IDF linear scoring (charge)	0.025
BM25 search, top-64 of 94,938	9.345
JCC search, top-64 of 94,938	1.409
14 kNN experts (both retrievers)	0.034
Gate features + mixture	0.019
Total	10.832

Adaptive fusion changes the weighting of retrieved evidence; computational savings occur only when confidence-based routing bypasses retrieval. The ACR gate cannot provide such savings by itself, because its features are computed from both retrieved lists. We therefore pair ACR with a routing rule that uses only the confidence of the base predictor, which is known before any search: confident cases keep the base prediction and bypass both retrievers, and the remaining cases are processed by ACR. Fig. 6 varies the routed fraction on the test split. Routing 10% of the charge cases already yields +0.15 points (all cases: +0.14); for articles, 20% yields +0.61 of the +0.62 points. For terms the best trade-off routes 50.0% of the cases and gains +1.48 points, more than routing every case (+1.28), because precedents occasionally overturn correct confident predictions. In deployment the threshold would be fixed on validation data, as for the confidence-triggered baseline in Table II. Table V lists the per-case cost of the full pipeline, without routing, on one CPU core. BM25 search over the 94,938 precedents dominates (9.3 ms), JCC search takes 1.4 ms, and the experts and gate add only hundredths of a millisecond. Only the routing rule changes this cost: when 10–30% of the charge and article cases are routed, the two searches run for that fraction alone, which removes 70–90% of the search cost, whereas ACR without routing always pays it in full.

5.7. Case Study

A test case reads, in translation: the defendant and the victim, fellow villagers with a long-standing family feud, met at a neighbour's wedding, where the defendant, drunk and in a fit of anger, slashed the victim's neck with a kitchen knife; forensic appraisal rated the wound as a minor injury (grade II), and the defendant fled before being arrested seven months later. The SVM predicts *intentional homicide* with probability 0.44, apparently misled by the knife-to-neck description. The nearest BM25 precedents are labelled intentional injury, intentional homicide and illegal intrusion into a residence, whereas the three nearest JCC precedents (cosine ≈ 0.81) are all *intentional injury*, the outcome that the injury grade supports. With weights 0.58 on the parametric expert and 0.36 on the JCC experts, the mixture predicts intentional injury, which is correct. Over the whole test split, ACR corrects 263, 401 and 1908 base errors while introducing 232, 266 and 1627 new ones for charges, articles and terms. Retrieved precedents also give the user something concrete to inspect: the decision can be checked against actual prior judgments rather than against attention weights.

6. Discussion

The experiments support three conclusions. First, precedents add information that a strong lexical classifier lacks, but only when they are retrieved in a judgment-aligned space or, for sentencing, matched lexically on case details. Second, a learned gate over retrievers and depths can stand in for a searched fixed configuration: it matches the best static accuracy in distribution without being uniformly better, and for sentencing it made better use of newly added precedents. Third, per-case adaptivity offers headroom beyond the current features. The oracle gap (94.3% versus 89.5% for charges and 73.0% versus 44.0% for terms) shows that the experts disagree usefully, so richer case-level signals can unlock further gains. LegalReasoner attributes 18% of its ECHR errors and 20% of its CAIL2018 errors to insufficient context [13]; our results suggest that precedents address this category mainly for sentencing, and that case-level legal signals, such as the constituent elements of an offence or the sentencing circumstances used by NeurJudge [10], are promising gate inputs.

Several extensions follow from these findings. The parametric expert is a linear model chosen to fit a single-CPU budget; because ACR only requires class probabilities, neural or LLM backbones [11], [21] can be wrapped unchanged, and the same protocol can be applied to other jurisdictions such as the ECHR setting studied in [4]. Rare charges call for retrieval strategies that pool evidence across related offences. Finally, because voting over historical decisions can reproduce past inconsistencies or biases, systems of this kind should support rather than replace judicial reasoning, and their retrieved precedents should be shown to the user.

7. Conclusion

We presented adaptive case retrieval, a retrieval-augmented approach to legal judgment prediction that mixes a calibrated parametric predictor with precedent votes from a judgment-consistent dense retriever and a lexical retriever at several depths, using a gate trained by maximum likelihood on held-out data. On CAIL2018-Small, precedents improve law-article and prison-term prediction significantly, the judgment-consistent retriever is essential for charges and articles, and the gate learns this division of labour by itself. In distribution it matches the best tuned static fusion, and for prison terms it keeps improving as the memory grows without re-training. Adaptive fusion changes the weighting of retrieved evidence rather than the amount of computation; the savings come from confidence-based routing, which bypassed retrieval for most charge and article cases at no cost in accuracy. Future work will add case-level legal features to the gate, balance precedent votes across classes, and evaluate

stronger backbones and further jurisdictions.

References

[1] Zhong, H., Xiao, C., Tu, C., Zhang, T., Liu, Z., & Sun, M. (2020). How does NLP benefit legal system: A summary of legal artificial intelligence. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL) (pp. 5218-5230). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.466

[2] Xiao, C., Zhong, H., Guo, Z., Tu, C., Liu, Z., Sun, M., Feng, Y., Han, X., Hu, Z., Wang, H., & Xu, J. (2018). CAIL2018: A large-scale legal dataset for judgment prediction. arXiv, arXiv:1807.02478. https://doi.org/10.48550/arXiv.1807.02478

[3] Aletras, N., Tsarapatsanis, D., Preoţiuc-Pietro, D., & Lampos, V. (2016). Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective. PeerJ Computer Science, 2, Article e93. https://doi.org/10.7717/peerj-cs.93

[4] Chalkidis, I., Androutsopoulos, I., & Aletras, N. (2019). Neural legal judgment prediction in English. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL) (pp. 4317-4323). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1424

[5] Luo, B., Feng, Y., Xu, J., Zhang, X., & Zhao, D. (2017). Learning to predict charges for criminal cases with legal basis. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 2727-2736). Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1289

[6] Zhong, H., Guo, Z., Tu, C., Xiao, C., Liu, Z., & Sun, M. (2018). Legal judgment prediction via topological learning. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 3540-3549). Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1390

[7] Ma, Y., Shao, Y., Wu, Y., Liu, Y., Zhang, R., Zhang, M., & Ma, S. (2021). LeCaRD: A legal case retrieval dataset for Chinese law system. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 2342-2348). ACM. https://doi.org/10.1145/3404835.3463250

[8] Shao, Y., Mao, J., Liu, Y., Ma, W., Satoh, K., Zhang, M., & Ma, S. (2020). BERT-PLI: Modeling paragraph-level interactions for legal case retrieval. In Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI) (pp. 3501-3507). https://doi.org/10.24963/ijcai.2020/484

[9] Xu, N., Wang, P., Chen, L., Pan, L., Wang, X., & Zhao, J. (2020). Distinguish confusing law articles for legal judgment prediction. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL) (pp. 3086-3095). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.280

[10] Yue, L., Liu, Q., Jin, B., Wu, H., Zhang, K., An, Y., Cheng, M., Yin, B., & Wu, D. (2021). NeurJudge: A circumstance-aware neural framework for legal judgment prediction. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 973-982). ACM. https://doi.org/10.1145/3404835.3462826

[11] Yue, X., & Zhu, R. (2024). An explainable hybrid machine learning framework for cash-flow-conditioned multi-horizon liquidity risk early warning: An SME-oriented benchmark study. World Journal of Management Science, 2(1), 63-72.

[12] Zhang, H., & Dou, Z. (2023). Case retrieval for legal judgment prediction in legal artificial intelligence. In Proceedings of the 22nd Chinese National Conference on Computational Linguistics (CCL) (pp. 801-812).

[13] Wang, X., Zhang, X., Hoo, V., Shao, Z., & Zhang, X. (2024). LegalReasoner: A multi-stage framework for legal judgment prediction via large language models and knowledge integration. IEEE Access, 12, 166843-166854. https://doi.org/10.1109/ACCESS.2024.3496666

[14] Mallen, A., Asai, A., Zhong, V., Das, R., Khashabi, D., & Hajishirzi, H. (2023). When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL) (pp. 9802-9822). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.546

[15] Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2024). Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In Proceedings of the International Conference on Learning Representations (ICLR).

[16] Jeong, S., Baek, J., Cho, S., Hwang, S. J., & Park, J. C. (2024). Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL) (pp. 7036-7050). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.naacl-long.389

[17] Zheng, X., Zhang, Z., Guo, J., Huang, S., Chen, B., Luo, W., & Chen, J. (2021). Adaptive nearest neighbor machine translation. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP) (pp. 368-374). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-short.47

[18] Zhou, F. (2026). Moment-constrained side-force allocation for wind-assisted merchant ships: Hydrodynamic performance of retractable anti-leeway fins and flap rudders. Innovation and Technology Studies, 3(1), 27-41.

[19] Yang, W., Jia, W., Zhou, X., & Luo, Y. (2019). Legal judgment prediction via multi-perspective bi-feedback network. In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI) (pp. 4085-4091). https://doi.org/10.24963/ijcai.2019/567

[20] Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., & Androutsopoulos, I. (2020). LEGAL-BERT: The muppets straight out of law school. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 2898-2904). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.findings-emnlp.261

[21] Xiao, C., Hu, X., Liu, Z., Tu, C., & Sun, M. (2021). Lawformer: A pre-trained language model for Chinese legal long documents. AI Open, 2, 79-84. https://doi.org/10.1016/j.aiopen.2021.06.003

[22] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-T. (2020). Dense passage retrieval for open-domain question answering. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) (pp. 6769-6781). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.550

[23] Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., & Krishnan, D. (2020). Supervised contrastive learning. In Advances in Neural Information Processing Systems 33 (NeurIPS) (pp. 18661-18673).

[24] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-T., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation

for knowledge-intensive NLP tasks. In Advances in Neural Information Processing Systems 33 (NeurIPS) (pp. 9459-9474).

[25] Khandelwal, U., Levy, O., Jurafsky, D., Zettlemoyer, L., & Lewis, M. (2020). Generalization through memorization: Nearest neighbor language models. In Proceedings of the International Conference on Learning Representations (ICLR).

[26] Khandelwal, U., Fan, A., Jurafsky, D., Zettlemoyer, L., & Lewis, M. (2021). Nearest neighbor machine translation. In Proceedings of the International Conference on Learning Representations (ICLR).

[27] Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning (ICML) (pp. 1321-1330). PMLR.

[28] Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333-389. https://doi.org/10.1561/1500000019

[29] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391-407. https://doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

[30] Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation, 10(7), 1895-1923. https://doi.org/10.1162/089976698300017197

[31] Yang, J.-S., Zeng, Z., & Shen, Z. (2025). Neural-symbolic dual-indexing architectures for scalable retrieval-augmented generation. IEEE Access, 13, 1-15. https://doi.org/10.1109/ACCESS.2025.3638761

[32] Rhee, M., Zhang, Y., Li, H., Wang, J., & Chen, X. (2026). Enhancing web search agents with self-play contrastive fine-tuning. IEEE Access, 14, 1-14. https://doi.org/10.1109/ACCESS.2026.3717594