PAGE: PATENT ASSESSMENT WITH LLM-EXTRACTED TECHNICAL FEATURE GRAPHS

Anonymous authors
Paper under double-blind review

ABSTRACT

General-purpose large language models (LLMs) have demonstrated remarkable capabilities in language understanding, reasoning and generation across a wide range of domains. Patent assessment, however, remains difficult for them, because they struggle to understand the complex structural information embedded in patent claims. We define *patent assessment* as three critical tasks that an examiner performs on a patent: novelty evaluation, prior-art retrieval and patent classification. Novelty evaluation is the most demanding of the three, because establishing a novelty conflict requires that every technical feature of the patent under examination, together with its properties, is covered by the prior art. Considering that patent claims are inherently written with a hierarchical structure, we propose **PAGE** (**Pa**tent **G**raph **E**xaminer), a two-stage framework that addresses all three tasks using a unified representation. PAGE first devises an LLM to convert claims and prior-art texts into directed technical feature graphs, whose nodes represent technical features and typed edges encode composition, interaction, taxonomy and property relations. A lightweight graph neural network then encodes each graph using frozen text embeddings of features and relations, edge-aware graph attention, and attentive readout. Task-specific heads map the resulting graph embeddings to predictions for the three tasks: a Siamese interaction head for novelty evaluation, a contrastively trained dual encoder for prior-art retrieval, and a multilayer perceptron classifier for patent classification. Experiments on PatentMatch and CLEF-IP datasets show that PAGE effectively improves patent novelty evaluation and is equally effective for prior-art retrieval and patent classification.

1 INTRODUCTION

In recent years, large language models (LLMs) have advanced at a remarkable pace. Advances in large-scale pre-training and instruction tuning have yielded models capable of following instructions, performing multi-step reasoning, and generating fluent text across diverse domains (Brown et al., 2020; Wei et al., 2022). This progress has also extended to the patent domain. Researchers have fine-tuned LLMs for invention concept and patent generation (Ren et al., 2025), investigated their ability to draft patent claims (Jiang et al., 2025) and methods for evaluating such claims (Yoo et al., 2025), developed multimodal models for describing patent figures (Shukla et al., 2025), used LLMs to extract technical contradictions from patents (Trapp & Warschat, 2024), and integrated LLMs with retrieval-augmented generation for patent retrieval and classification (Ding et al., 2025). Most of these efforts concentrate on generating or categorising patent text. Novelty evaluation, the decision at the heart of patent examination, has received far less attention, although it determines whether an invention can be protected at all. A claim lacks novelty only if a single prior-art document discloses every one of its technical features together with their properties (European Patent Office, 2024), and a passage that describes the same apparatus but omits a single restriction does not destroy novelty.

Only a few studies bring LLMs to this decision. Ikoma & Mitamura (2025) build a novelty evaluation dataset from real examination cases and let LLMs compare claims with cited prior art. Knappich et al. (2026) decompose claims into features and analyse each feature against prior-art passages with LLM workflows. Xiong et al. (2025) strengthen LLM-based patent matching on PatentMatch with entities and ontologies drawn from the parametric memory of the model. These three LLM-based approaches suffer from two limitations. First, they let the model reason over every claim–prior art

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

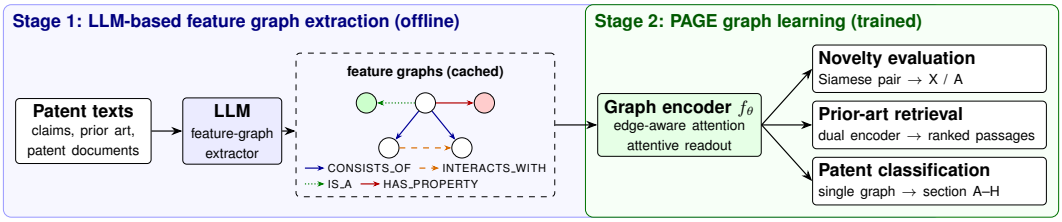


Figure 1: Overview of PAGE. Stage 1 converts each patent text once into a directed technical feature graph with four relation types and caches it. Stage 2 trains a shared graph encoder whose embeddings are used by three task heads for novelty evaluation, prior-art retrieval and patent classification.

pair, or over every feature–passage pair, at inference time, so that the decision rests on the ability of the model to align long and nested legal texts in a single reasoning process. In our experiments, open-weight LLMs queried in this way answer "no conflict" almost everywhere. Second, they represent a claim as flat text (Ikoma & Mitamura, 2025), as a list of features (Knappich et al., 2026) or as a set of categorical entities (Xiong et al., 2025), which leaves implicit the relations that give a claim its meaning: which component contains which, how components interact, and which properties restrict them. Outside the LLM setting, Schmitt et al. (2023) compare the feature combinations of claims and prior art with a mathematical-logical procedure. Their procedure states the relations between features explicitly, but the semantic structures on which it operates are constructed manually, and thus it does not scale to a large corpus.

To address these limitations, we propose PAGE (**Pa**tent **G**raph **E**xaminer) for patent assessment, a two-stage framework that separates reading a patent from judging it: an LLM reads every claim and every prior-art text once and converts it into a directed technical feature graph, and a lightweight graph neural network trained on these graphs performs the three assessment tasks, namely novelty evaluation, prior-art retrieval and assignment to sections of the International Patent Classification (IPC), as illustrated in Figure 1. This design follows the way claims are written. That is, a claim names an invention and introduces its elements with a transitional phrase such as "comprising". It then refines each element through "wherein" and "having" clauses. The content of a claim is thus a hierarchy of technical features connected by typed relations, which can be represented as a directed graph. Given that novelty is decided feature by feature, every node represents a technical feature, such as a component, a functional unit, a process step, a material, a signal or a parameter. The edge types reflect the ways in which a feature can be disclosed or missed: CONSISTS_OF captures the element hierarchy, so that a feature is matched in its proper context; INTERACTS_WITH captures functional couplings, so that the same components in a different arrangement remain distinguishable; IS_A links generic and specific terms, which matters when prior art discloses a genus and the claim recites a species; and HAS_PROPERTY captures the parametric and qualitative restrictions on which novelty frequently turns. PAGE asks an LLM to extract such a graph once per document, with the same schema for claims and prior art, and caches it. The LLM thus contributes its reading ability, while the comparison is performed by a trained graph model, in line with recent evidence that explicit structure benefits patent analysis (Gao et al., 2024) and that LLM-derived graphs support reliable evaluation with lightweight graph models (Feng et al., 2025). Across five open-weight LLMs, using the model as the feature-graph extractor of PAGE yields substantially better patent novelty evaluation than using the same model as a judge, and PAGE also outperforms supervised text baselines trained on the same data.

The graph neural network of PAGE is built around the comparison that novelty requires (Figure 3). Feature texts and relation phrases are embedded by a frozen sentence encoder, and each edge additionally carries its relation type. Edge-aware graph attention lets both the type and the wording of a relation determine how strongly a feature is influenced by its neighbours, and an attentive readout concentrates the graph embedding on the few features that distinguish one disclosure from another. One encoder serves three task heads, each trained with a loss suited to its task. For novelty evaluation, the claim and the prior art are encoded with shared weights, and an interaction head over the two embeddings models the directional question of whether the prior art covers the claim, which is trained with binary cross-entropy. For prior-art retrieval, the encoder acts as a dual encoder trained with a symmetric InfoNCE loss. In-batch passages teach the model to push unrelated documents away and A-category citations act as hard negatives that retain the fine-grained novelty distinction.

Document embeddings are indexed once and reused for every query. For patent classification, a two-layer MLP head on a single graph embedding is trained with class-weighted cross-entropy to handle the imbalanced distribution of IPC sections.

In summary, our contributions are as follows.

- We propose PAGE, a two-stage framework for patent assessment in which an LLM converts patent claims and prior art into directed technical feature graphs and a lightweight graph neural network makes the decision.
- We design a graph encoder with edge-aware attention and attentive readout that is shared by three task heads with dedicated objectives for novelty evaluation, prior-art retrieval and patent classification.
- Experiments on PatentMatch and CLEF-IP datasets with five open-weight LLMs, under a strict partition that shares neither claims nor passages across splits, show that PAGE substantially improves patent novelty evaluation and is effective for prior-art retrieval and patent classification.

2 Related Work

LLMs for patent generation and analysis. Recent work applies LLMs across the patent life cycle. PatentGPT (Ren et al., 2025) injects domain knowledge through continued pre-training, supervised fine-tuning and reinforcement learning from human feedback to generate invention concepts and patents. Jiang et al. (2025) construct a benchmark for claim generation and find that LLMs write strong first independent claims, with more room for improvement on dependent claims, whose meaning hinges on their relation to earlier claims. PatentScore (Yoo et al., 2025) evaluates generated claims through a hierarchical decomposition of claim elements and legal validation patterns. PatentLMM (Shukla et al., 2025) combines a patent-specific vision encoder with a domain-adapted LLM to describe patent figures. Trapp & Warschat (2024) extract TRIZ contradictions from patents through prompt engineering with GPT-4. Ding et al. (2025) integrate LLM embeddings with retrieval-augmented generation for patent retrieval and classification. Several of these studies point to the hierarchical organisation of claims as a defining property of patent text, but they exploit this property only inside the prompt of a generative model.

Patent novelty evaluation and matching. The PatentMatch dataset (Risch et al., 2020) casts novelty-relevant matching as classifying claim–passage pairs from examiner citations into novelty-destroying (X) and background (A) references, and sentence encoders adapted to patents, such as PatentSBERTa (Bekamiri et al., 2024), are widely used for patent similarity. Ikoma & Mitamura (2025) introduce a novelty evaluation dataset derived from examination cases and show that generative LLMs can compare claims with cited prior art and explain their decisions. Knappich et al. (2026) move to the feature level with FiNE-Patents, where LLM workflows decompose a claim into features, retrieve the prior-art passages that disclose each feature and derive a claim-level decision. MemGraph (Xiong et al., 2025) prompts LLMs to recall entities and ontologies of patents and uses them to improve patent matching on PatentMatch. Schmitt et al. (2023) formalise patentability as a logical comparison of feature combinations. For patent approval prediction, Gao et al. (2024) show that a fine-grained claim dependency graph outperforms scaled-up language models. These works still lack an explicit representation of technical features and their relations that could be compared by a trained model rather than in a single LLM pass. Appendix A reviews these approaches in detail.

3 Method

3.1 Overview

PAGE performs patent assessment in two stages (Figure 1): an offline extraction stage, in which an LLM $\mathcal{E}_{\text{LLM}}$ reads a patent text t once and converts it into a technical feature graph $G = (V, E)$ that is cached, and a trained learning stage, in which a graph encoder f_θ maps G to an embedding $\mathbf{h}_G \in \mathbb{R}^d$ and a task head g_ψ turns one or more such embeddings into the prediction of one of the

three tasks:

$$t \xrightarrow{\mathcal{E}_{\text{LLM}}} G = (V, E) \xrightarrow{f_\theta} \mathbf{h}_G \xrightarrow{g_\psi} \hat{y}. \tag{1}$$

The extractor is used without fine-tuning, and only θ and ψ are trained. Appendix C gives the method pseudocode.

3.2 Technical feature graph extraction

Schema. A graph consists of nodes $v_i = (\texttt{id}, \texttt{subject}, \texttt{text})$ and directed, typed edges $e_{ij} = (\texttt{source}, \texttt{target}, r_{ij}, \texttt{text})$ with

$$r_{ij} \in \mathcal{R} = \{\text{CONSISTS_OF},\ \text{INTERACTS_WITH},\ \text{IS_A},\ \text{HAS_PROPERTY}\}.$$

Each node is a technical feature, whose `subject` holds its canonical name and whose `text` holds the span from which it was derived, so that the node is interpretable on its own. Each edge stores its relation type and the phrase that expresses the relation in the source text. Figure 2 shows an extracted graph.

Extraction. The LLM receives the text together with the schema and returns the graph as JSON (Appendix D). The same prompt is used for claims and for prior-art passages, so that both inputs of a pair are represented in the same space and the extractor can be reused on any patent corpus. Each unique text is extracted once, keyed by a hash of its normalised form, and the validated output is cached. Training and inference of the graph neural network read directly from this cache.

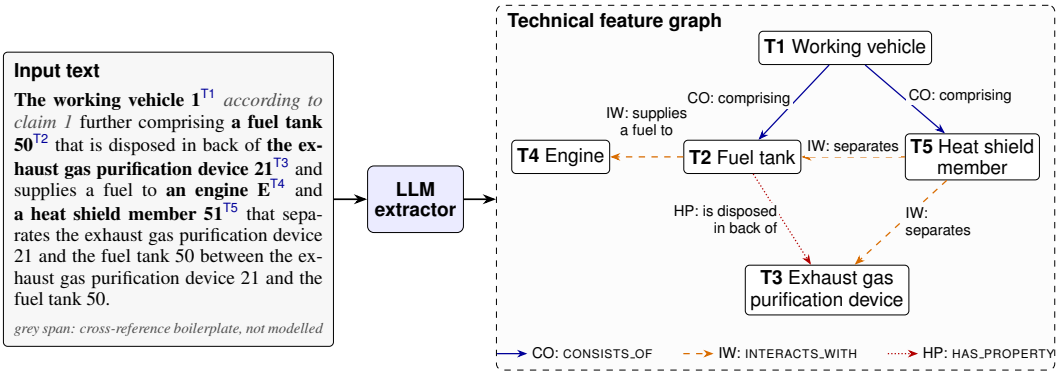


Figure 2: Technical feature graph extraction for a PatentMatch claim fragment. The LLM identifies technical features (bold spans, indexed T1–T5), skips the cross-reference boilerplate, and returns nodes and typed edges as JSON. Each node stores a canonical name and its source span, and each edge stores its relation type and the phrase that expresses it, so that the two INTERACTS_WITH edges of T5 and the HAS_PROPERTY edge that places the fuel tank behind the purification device remain distinguishable.

3.3 Node and edge features

Let $\phi(\cdot)$ denote a frozen sentence encoder with output dimension 384 (all-MiniLM-L6-v2; Wang et al., 2020; Reimers & Gurevych, 2019). Node and edge features are

$$\mathbf{x}_i = \phi\big(\text{txt}(v_i)\big) \in \mathbb{R}^{384}, \qquad \mathbf{e}_{ij} = \big[\,\text{onehot}(r_{ij})\,;\,\phi\big(\text{txt}(e_{ij})\big)\,\big] \in \mathbb{R}^{388}, \tag{2}$$

where $\text{txt}(\cdot)$ is the textual content of a node or an edge. Each edge thus carries both its relation type and the phrase that licensed it, so that, for instance, "separates" and "supplies a fuel to" remain distinguishable although both are INTERACTS_WITH relations.

3.4 Graph encoder

The encoder (Figure 3, top) stacks two TransformerConv layers (Shi et al., 2021), which incorporate edge features into both the attention scores and the messages. For node i with incoming neighbours

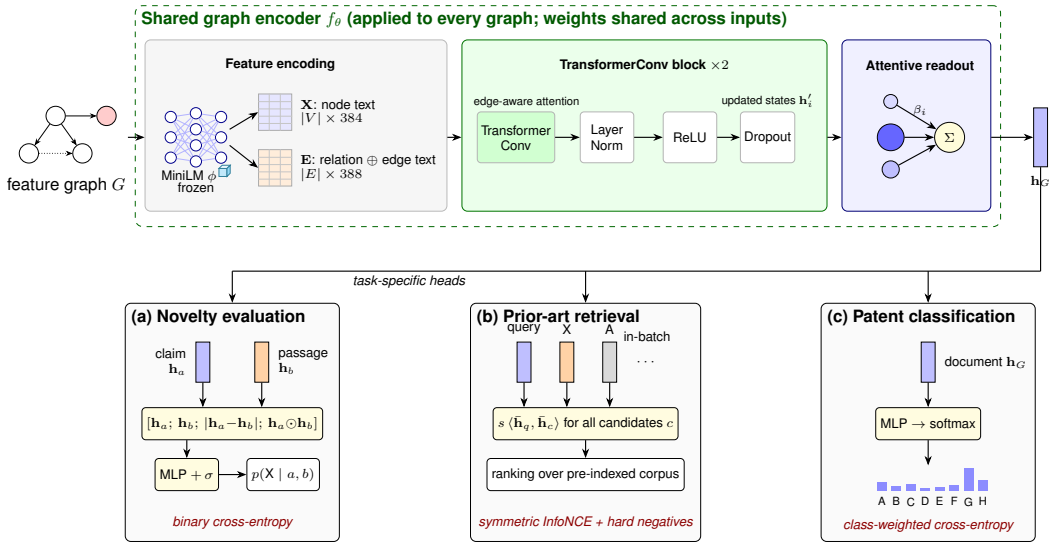


Figure 3: The graph neural network of PAGE. The shared encoder reads a feature graph in three steps: a frozen sentence encoder ( ) turns node texts and relation phrases into the feature matrices $\mathbf{X}$ and $\mathbf{E}$; two TransformerConv blocks update the node states, each consisting of a TransformerConv layer whose attention score α_{ij} is conditioned on the edge feature $\mathbf{e}_{ij}$, followed by layer normalisation, ReLU and dropout; and the attentive readout sums the node states with weights β_i, shown as node size, into $\mathbf{h}_G$. (a) For novelty evaluation, claim and passage embeddings are compared by an interaction head trained with binary cross-entropy. (b) For prior-art retrieval, the encoder acts as a dual encoder trained with symmetric InfoNCE over in-batch and A-category negatives. (c) For patent classification, a multilayer perceptron (MLP) head is trained with class-weighted cross-entropy.

$\mathcal{N}(i)$,

$$\alpha_{ij} = \mathrm{softmax}_{j \in \mathcal{N}(i)} \left(\frac{(W_q \mathbf{h}_i)^\top (W_k \mathbf{h}_j + W_e \mathbf{e}_{ij})}{\sqrt{d_k}} \right), \tag{3}$$

$$\mathbf{h}'_i = W_r \mathbf{h}_i + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} \, (W_v \mathbf{h}_j + W_e \mathbf{e}_{ij}). \tag{4}$$

The first layer concatenates four attention heads and the second uses a single head. Each layer is followed by layer normalisation, ReLU and dropout. Since the edge feature $\mathbf{e}_{ij}$ enters the attention score, the influence of a neighbour j on a feature i depends not only on the two features themselves but also on the type and the wording of the relation that connects them.

Attentive readout. The graph embedding is an attention-weighted sum of node states (Li et al., 2016),

$$\beta_i = \mathrm{softmax}_{\, i \in V} \big(a(\mathbf{h}_i) \big), \qquad \mathbf{h}_G = \sum_{i \in V} \beta_i \mathbf{h}_i, \tag{5}$$

with a small scoring network $a(\cdot)$. Novelty typically hinges on a few distinguishing features, and the readout learns to give them large weights instead of averaging them with the remaining nodes.

3.5 Task heads and objectives

Novelty evaluation. The claim graph G_a and the passage graph G_b are encoded with shared weights, and the head predicts

$$p(\mathrm{X} \mid a, b) = \sigma\Big(\mathrm{MLP}\big([\mathbf{h}_a; \, \mathbf{h}_b; \, |\mathbf{h}_a - \mathbf{h}_b|; \, \mathbf{h}_a \odot \mathbf{h}_b] \big) \Big) \tag{6}$$

and is trained with binary cross-entropy. The difference and product terms expose dimension-wise agreement and disagreement, and the ordered concatenation lets the head represent the directional relation "b discloses the features of a".

Prior-art retrieval. For retrieval over a corpus, the encoder serves as a dual encoder (Karpukhin et al., 2020) with L_2-normalised embeddings $\bar{\mathbf{h}}$, so that document embeddings are computed once and indexed. Given B query claims with positive passages $\{p_k\}_{k=1}^{B}$ and a set N_B of hard negatives, the query-side loss is

$$\ell_q = -\log \frac{\exp\left(s\,\langle \bar{\mathbf{h}}_q, \bar{\mathbf{h}}_{p_q}\rangle\right)}{\sum_{c\in\{p_k\}_{k=1}^{B}\cup N_B} \exp\left(s\,\langle \bar{\mathbf{h}}_q, \bar{\mathbf{h}}_c\rangle\right)}, \tag{7}$$

which is averaged with the symmetric passage-to-query term (van den Oord et al., 2018), and we use $s = 20$. Hard negatives are A-category passages cited against the same claim, and thus the retriever learns the fine-grained X/A distinction as part of its ranking objective.

Patent classification. A two-layer MLP g_ψ, with a hidden layer of the same width as $\mathbf{h}_G$, ReLU and dropout, maps the graph embedding $\mathbf{h}_G$ to C logits and is trained with cross-entropy weighted by $w_c = n/(C\,n_c)$, where n_c is the number of training documents of class c, to balance the IPC sections.

4 Experimental Setup

4.1 Datasets

PatentMatch. We use a balanced file of the PatentMatch release (Risch et al., 2020) with 69576 claim–passage pairs and an X:A ratio of $1{:}1$. Pairs are partitioned by the connected components of the bipartite graph between claim texts and passage texts, so that no claim and no passage appears in more than one of the training, validation and test partitions ($48704/10436/10436$ pairs). For retrieval, claims serve as queries, the union of all cited passages forms the corpus ($|C| = 11053$), and the X-category passages of a claim are its relevant documents. Queries are partitioned in the same way.

CLEF-IP. The CLEF-IP 0.54M corpus (Kamateri et al., 2022), derived from CLEF-IP (Piroi et al., 2011), contains about 0.54 million patents labelled with their main IPC subclass. We map subclasses to the eight IPC sections A–H and use a class-stratified sample of 20000 documents with a stratified $70/15/15$ split.

4.2 Baselines

Novelty evaluation. In *direct LLM judgement*, five models , namely Qwen2.5-7B, Qwen2.5-14B, Llama-3.1-8B, Gemma-2-9B and Phi-4 (14B), decide X/A directly from the claim and the passage with two in-context examples. Appendix B details their configurations. The prompt is given in Appendix D. The same five models serve as extractors in PAGE. The text baselines are cosine similarity of frozen MiniLM embeddings, logistic regression on $[|\mathbf{a}-\mathbf{b}|; \mathbf{a}\odot\mathbf{b}]$ over the same embeddings, and a BERT cross-encoder (Devlin et al., 2019) fine-tuned on the training pairs.

Prior-art retrieval. We compare with BM25 (Robertson & Zaragoza, 2009), with MiniLM and PatentSBERTa in the zero-shot setting, and with PatentSBERTa fine-tuned with the same objective, effective batch size (64), hard negatives, epoch budget and model-selection rule as PAGE, using gradient caching (Gao et al., 2021).

Patent classification. Qwen2.5-7B and Qwen2.5-14B, the two extractors used by PAGE on this task, assign the IPC section directly from the document text with three in-context examples, and the section label is parsed from the generated output. The supervised text baseline is all-MiniLM-L6-v2 fine-tuned with a linear classification head on the training documents.

Table 1: Novelty evaluation on PatentMatch. For each LLM, *Direct* lets the LLM output the X/A decision, and *PAGE* uses the same LLM only to extract the feature graphs, on which the graph neural network of PAGE makes the decision (the LLM itself is not fine-tuned). Acc.: accuracy; Prec.: precision; Rec.: recall. The best results in bold.

LLM / Method	Usage	Acc.	Prec.	Rec.	F1
Qwen2.5-7B	Direct	0.51	0.58	0.12	0.20
	PAGE	0.72	0.74	0.68	0.71
Llama-3.1-8B	Direct	0.51	0.59	0.11	0.19
	PAGE	0.73	0.75	0.69	0.72
Gemma-2-9B	Direct	0.52	0.60	0.13	0.21
	PAGE	0.71	0.72	0.68	0.70
Phi-4 (14B)	Direct	0.53	0.61	0.15	0.24
	PAGE	0.78	0.79	0.70	0.74
Qwen2.5-14B	Direct	0.54	0.63	0.14	0.23
	PAGE	**0.78**	**0.81**	**0.73**	**0.77**
MiniLM cosine		0.51	0.52	0.50	0.51
MiniLM + logistic regression		0.64	0.65	0.61	0.63
BERT cross-encoder (fine-tuned)		0.70	0.72	0.69	0.70

5 RESULTS

5.1 NOVELTY EVALUATION

In Table 1, the two rows of each LLM use the same weights and the same claim–passage pairs and differ only in whether the model emits the X/A verdict itself or the feature graph on which PAGE decides.

Used as judges, the five models answer with a pronounced prior towards "no conflict". Recall stays between 0.11 and 0.15 while precision reaches 0.58–0.63, which on a test set with an X:A ratio of $1{:}1$ implies that only about one pair in ten receives an X verdict. The issued verdicts are thus better than chance, but most novelty-destroying passages are passed over, and the accuracies of 0.51–0.54 follow from answering "A" almost everywhere. In examination this is the costly direction of the error, since a missed X reference lets an unpatentable claim through.

With PAGE, the same models reach accuracies of 0.71–0.78 and F1 scores of 0.70–0.77. The movement is carried by recall that rises to 0.68–0.73, while precision rises with it to 0.72–0.81, and thus detection and selectivity jointly improve rather than being traded against each other. A verdict emitted directly rests on a single pass over two long and nested legal sentences, whereas the feature graph states every feature, property and relation as a separate typed unit, and thus coverage is examined feature by feature.

The text baselines separate the contribution of the graph from that of the sentence encoder. PAGE builds its node and edge features from frozen MiniLM embeddings, and the two MiniLM baselines bound what those embeddings alone provide: 0.51 F1 score with cosine similarity and 0.63 F1 score with a supervised interaction classifier. Arranging the identical embeddings as typed feature graphs adds a further 0.14 F1 score, and the best configuration, with the Qwen2.5-14B extractor (0.78 accuracy, 0.77 F1 score), also leads the BERT cross-encoder fine-tuned on the same pairs (0.70/0.70).

Figure 4 repeats the comparison without a decision threshold. The AUC of direct judgement stays in 0.51–0.55 and is flat in model size, while the AUC obtained from the same models as extractors is 0.74–0.82 and grows with model capability. Naming the technical features of a text is already within reach of the 7B–9B models, which recover most of the improvement, whereas the ability to resolve a claim–passage pair into a verdict does not emerge when scaling from 7B to 14B. PAGE asks the LLM for the first ability and assigns the second to a trained graph model, so that a compact open-weight extractor suffices.

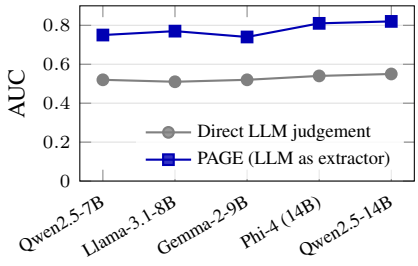


Figure 4: AUC of the same five LLMs used as direct judges and as feature-graph extractors in PAGE.

Table 2: Ablations of PAGE on novelty evaluation (Qwen2.5-14B extractor). Each row replaces one component.

Variant	Acc.	Prec.	Rec.	F1
PAGE (full)	**0.78**	**0.81**	**0.73**	**0.77**
TransformerConv → R-GCN	0.74	0.75	0.73	0.74
attentive readout → mean	0.76	0.76	0.72	0.74
interaction head → cosine	0.72	0.73	0.71	0.72

Table 3: Prior-art retrieval on the PatentMatch-derived benchmark over the full corpus. All methods use the same queries, corpus and relevance judgements.

Method	MAP	MRR	nDCG@10	Recall@10	Recall@20
BM25	0.154	0.302	0.209	0.202	0.264
MiniLM (zero-shot)	0.184	0.361	0.244	0.221	0.298
PatentSBERTa (zero-shot)	0.192	0.322	0.261	0.220	0.290
PatentSBERTa (fine-tuned)	0.445	0.576	0.486	0.442	0.578
PAGE	**0.641**	**0.695**	**0.662**	**0.567**	**0.713**

5.2 Ablation study

Table 2 replaces one component at a time while the Qwen2.5-14B extractor, the cached graphs, the data partition and the training budget stay fixed. Replacing TransformerConv by R-GCN (Schlichtkrull et al., 2018), which conditions on relation types but not on relation phrases, costs 0.03 F1 score: "separates" and "supplies a fuel to" are both INTERACTS_WITH, and only the phrase tells them apart. Mean pooling in place of the attentive readout costs 0.03 F1 and 0.05 precision, in line with its role of concentrating the graph embedding on the few distinguishing features. The largest effect comes from the interaction head: scoring the two graph embeddings by cosine similarity costs 0.06 accuracy and 0.05 F1 score, since novelty is a directional coverage relation that a symmetric score cannot express.

5.3 Prior-art retrieval

Retrieval moves the comparison from a single pair to a corpus of 11053 passages, and Table 3 shows PAGE ahead on all five metrics. The controlled comparison is with fine-tuned PatentSBERTa. It is trained with the same InfoNCE objective, the same A-category hard negatives and the same effective batch size, epoch budget and model-selection rule as PAGE, and thus the only difference between the two systems is the representation that is ranked. Under this protocol, PAGE adds 0.196 MAP (+44% relative) and 0.176 nDCG@10 (+36%). The lexical and zero-shot dense systems (0.154–0.192 MAP) indicate what makes this benchmark demanding: the candidates are all cited against the same claims, so relevant and non-relevant passages share topic and vocabulary and differ only in which technical features they disclose. The feature graph makes that difference the object of the comparison, and A-category hard negatives turn the X/A distinction into the ranking criterion. In practical terms, an MRR of 0.695 places the first novelty-destroying passage within the top two results on average, and Recall@20 of 0.713 puts about seven in ten of them before an examiner who inspects twenty passages per claim. Because graphs are encoded once and searched by inner product, the approach scales to the full corpus.

5.4 Patent classification

Table 4 reports IPC section classification on CLEF-IP. PAGE with the Qwen2.5-14B extractor reaches the best accuracy (0.645), macro-precision (0.667) and macro-F1 score (0.628) of the com-

Table 4: IPC section classification on CLEF-IP (eight sections A–H). *In-context* lets the LLM assign the section directly with three demonstrations. For PAGE, the same LLM only extracts the feature graphs and an MLP head on the graph embedding predicts the section. Acc.: accuracy; Macro-P and Macro-F1 average precision and F1 over the eight sections.

Method	LLM	Acc.	Macro-P	Macro-F1
In-context LLM	Qwen2.5-7B	0.475	0.597	0.465
	Qwen2.5-14B	0.550	0.642	0.546
MiniLM (fine-tuned)	–	0.633	0.633	0.625
PAGE	Qwen2.5-7B	0.590	0.631	0.573
	Qwen2.5-14B	**0.645**	**0.667**	**0.628**

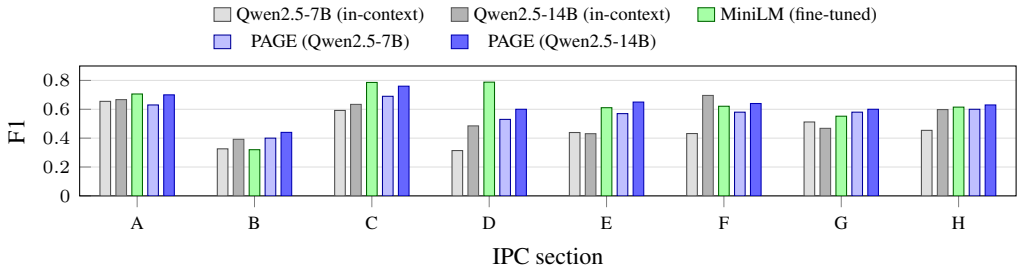


Figure 5: Per-section F1 on CLEF-IP. A: human necessities; B: performing operations, transporting; C: chemistry, metallurgy; D: textiles, paper; E: fixed constructions; F: mechanical engineering, lighting, heating, weapons, blasting; G: physics; H: electricity.

pared systems. The comparison with in-context classification is the counterpart of the pairwise experiment: with the same extractor, PAGE adds 0.095 accuracy and 0.082 macro-F1 score over the 14B model prompted to assign the section itself, and with the 7B extractor it adds 0.115 accuracy over the 7B model. The graphs are those the extraction stage has already produced and the classifier is a small two-layer MLP, so that one representation serves all three tasks.

PAGE and the MiniLM encoder fine-tuned end to end for this task occupy the same range (0.645/0.628 against 0.633/0.625), and Figure 5 shows that they reach it in different ways. An IPC section is announced largely by terminology, while the four relations of the schema are class-independent by design. As a result, the structural channel is close to uniform across sections and the discriminative signal is carried by the feature texts, the same channel that the fine-tuned encoder reads. What differs is the distribution over sections. Per-section F1 runs from 0.44 to 0.76 for PAGE (standard deviation 0.087) against 0.32 to 0.79 for the fine-tuned encoder (0.140). PAGE raises the weakest section B by 0.12 F1 and leads on E, F, G and H, while the text encoder peaks on the sections C and D with the most distinctive vocabulary. A classifier that serves all eight sections is judged by its weakest one, and on that criterion the graph representation is the more even of the two at equal or better average performance.

6 Conclusion

In this paper, we propose PAGE, a two-stage framework for patent assessment in which an LLM converts patent claims and prior-art texts into directed technical feature graphs. On these graphs, a lightweight graph neural network with edge-aware attention and attentive readout learns feature-level comparisons for novelty evaluation, prior-art retrieval, and patent classification. Mirroring the hierarchical way claims are written, the graphs make technical features, their properties and their relations explicit. Across five open-weight LLMs, PAGE raises novelty evaluation on PatentMatch well above direct LLM judgement and supervised text baselines. The same graphs deliver strong prior-art retrieval, and a lightweight MLP head extends the framework to patent classification. Explicit feature graphs thus connect the reading ability of LLMs with the feature-level comparisons that patent examination requires.

AI USE STATEMENT

In this work, we used generative AI tools for checking grammar. We have not used generative AI tools for retrieval and discovery, research ideation or execution, drafting sections of the paper, generating synthetic datasets, or proving mathematical claims, and other potential AI tasks are not applicable to this work. We have reviewed all AI-assisted work. All LLM-suggested edits are carefully checked and manually verified by the authors to ensure accuracy and clarity. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses publicly available patent documents and examiner citations. The proposed methods are designed to support patent professionals in prior-art search and novelty evaluation.

REPRODUCIBILITY STATEMENT

Section 3.2 and Appendix D specify the extraction schema, prompts and caching, and §4 describes the data partitions. We will release the prompts, the cached graphs, the partition indices, and the training and evaluation code.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, et al. Phi-4 technical report, 2024.

Hamid Bekamiri, Daniel S. Hain, and Roman Jurowetzki. PatentSBERTa: A deep NLP based hybrid model for patent distance and classification using augmented SBERT. *Technological Forecasting and Social Change*, 206:123536, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, volume 33, pp. 1877–1901, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pp. 4171–4186, 2019.

Yao Ding, Yuqing Wu, and Ziyang Ding. An automatic patent literature retrieval system based on LLM-RAG, 2025.

European Patent Office. Guidelines for examination in the European Patent Office, Part G, Chapter VI: Novelty. `https://www.epo.org/en/legal/guidelines-epc`, 2024.

Tao Feng, Yihang Sun, and Jiaxuan You. GraphEval: A lightweight graph-based LLM framework for idea evaluation. In *ICLR*, 2025.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size under memory limited setup. In *RepL4NLP*, pp. 316–321, 2021.

Xiaochen Kev Gao, Feng Yao, Kewen Zhao, Beilei He, Animesh Kumar, Vish Krishnan, and Jingbo Shang. Beyond scaling: Predicting patent approval with domain-specific fine-grained claim dependency graph. In *NAACL*, 2024.

Gemma Team. Gemma 2: Improving open language models at a practical size, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024.

Hayato Ikoma and Teruko Mitamura. Can AI examine novelty of patents? novelty evaluation based on the correspondence between patent claim and prior art, 2025.

Lekang Jiang, Caiqi Zhang, Pascal A. Scherz, and Stefan Goetz. Can large language models generate high-quality patent claims? In *NAACL*, pp. 1272–1287, 2025.

Eleni Kamateri, Vasileios Stamatis, Konstantinos Diamantaras, and Michail Salampasis. CLEFIP-0.54M: A dataset for patent classification. `https://github.com/ekamater/CLEFIP-0.54M`, 2022.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, pp. 6769–6781, 2020.

Valentin Knappich, Anna Hätty, Simon Razniewski, and Annemarie Friedrich. Is it novel and why? fine-grained patent novelty prediction based on passage retrieval. In *SIGIR*, pp. 845–856, 2026.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. In *ICLR*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Florina Piroi, Mihai Lupu, Allan Hanbury, and Veronika Zenz. CLEF-IP 2011: Retrieval in the intellectual property domain. In *CLEF Labs and Workshops, Notebook Papers*, 2011.

Qwen Team. Qwen2.5 technical report, 2025.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *EMNLP-IJCNLP*, pp. 3982–3992, 2019.

Runtao Ren, Jian Ma, and Jianxi Luo. Large language model for patent concept generation. *Advanced Engineering Informatics*, 65:103301, 2025.

Julian Risch, Nicolas Alder, Christoph Hewel, and Ralf Krestel. PatentMatch: A dataset for matching patent claims and prior art, 2020.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *ESWC*, pp. 593–607, 2018.

Valentin J. Schmitt, Lothar Walter, and Frank C. Schnittker. Assessment of patentability by means of semantic patent analysis: A mathematical-logical approach. *World Patent Information*, 73: 102182, 2023.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *IJCAI*, pp. 1548–1554, 2021.

Shreya Shukla, Nakul Sharma, Manish Gupta, and Anand Mishra. PatentLMM: Large multimodal model for generating descriptions for patent figures. In *AAAI*, 2025.

Stefan Trapp and Joachim Warschat. LLM-based extraction of contradictions from patents. In *TFC*, pp. 3–19, 2024.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *NeurIPS*, volume 33, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, volume 35, 2022.

Qiushi Xiong, Zhipeng Xu, Zhenghao Liu, Mengjia Wang, Zulong Chen, Yue Sun, Yu Gu, Xiaohua Li, and Ge Yu. Enhancing the patent matching capability of large language models via the memory graph. In *SIGIR*, pp. 337–347, 2025.

Yongmin Yoo, Qiongkai Xu, and Longbing Cao. PatentScore: Multi-dimensional evaluation of LLM-generated patent claims. In *EMNLP*, pp. 30715–30734, 2025.

A Detailed Related Work

This appendix expands Section 2 with the method details that space did not allow there. Table 5 summarises the methods that address the novelty decision itself along three axes: how a claim is represented, how the decision is produced, and which tasks the method covers.

A.1 LLMs for patent generation, drafting and evaluation

PatentGPT (Ren et al., 2025) adapts a general-purpose LLM to the patent domain with knowledge-injection pre-training, domain-specific supervised fine-tuning and RLHF, and uses the result to generate invention concepts and the patent text that protects them. Jiang et al. (2025) build a claim-generation benchmark and report that claims generated from a description outperform those generated from an abstract, that patent-specific LLMs trail strong general-purpose ones, and that quality drops sharply from the first independent claim to the dependent claims, whose meaning depends on earlier claims. PatentScore (Yoo et al., 2025) evaluates generated claims by first decomposing them hierarchically into elements and then applying legally and technically grounded validation patterns, reaching a stronger correlation with expert judgement than standard NLG metrics on 400 first claims. PatentLMM (Shukla et al., 2025) pairs a patent-specific vision encoder with a domain-adapted LLM to describe patent figures, using the PatentDesc-355K corpus of about 355K figures drawn from over 60K US patents. Trapp & Warschat (2024) use a single GPT-4 prompt to detect, extract and summarise TRIZ contradictions in patent text and assign them to the 39 TRIZ engineering parameters, reaching a BERTScore F1 of 0.93. Ding et al. (2025) combine LLM embeddings with retrieval-augmented generation for patent search and classification. Together these studies show that a general-purpose LLM can reliably name technical content once the output format is specified precisely, which is the ability the extraction stage of PAGE relies on, but each of them keeps that ability inside a generation or scoring pipeline rather than exposing it as a reusable structured representation.

A.2 Novelty evaluation and patent matching

PatentMatch (Risch et al., 2020) is built from EPO search reports, in which examiners cite prior-art paragraphs against a claim and mark each citation X (novelty-destroying on its own) or A (background only); pairing every citation with its claim gives over six million pairs, released in a balanced 1:1 X:A file. Because the X and A passages of a claim are cited against the same claim, they share topic and vocabulary and differ only in which features they disclose, which is what makes the task hard. PatentSBERTa (Bekamiri et al., 2024) adapts Sentence-BERT to claims with an augmented-SBERT training procedure and is widely used as an off-the-shelf 768-dimensional patent encoder; we use it both zero-shot and fine-tuned. On the decision itself, Ikoma & Mitamura (2025) introduce the first novelty-evaluation dataset from real examination cases and show that generative LLMs judge better than classifiers, though still from a single reasoning pass over claim and prior art. Knappich et al. (2026) argue that a claim-level verdict is too coarse and introduce FiNE-Patents (3,658 claims with feature-level references from European Search Opinions), reformulating the task as retrieving, for every feature, the passages that disclose it; the resulting LLM workflow outperforms embedding baselines but analyses every feature of every claim with a separate LLM call at inference time. MemGraph (Xiong et al., 2025) instead prompts an LLM to recall entities and their ontology labels from its own parametric memory to support matching on PatentMatch, a representation of categorised entities that leaves out relations such as composition and interaction. Schmitt et al. (2023) formalise patentability as a mathematical-logical comparison of manually built feature combinations, which is explicit and auditable but does not scale beyond hand-constructed cases.

Table 5: Methods that address the novelty decision, compared along the representation of a claim, the mechanism that produces the decision, and the tasks covered. The last row is the framework proposed here.

Method	Claim representation	Decision mechanism	Tasks
Ikoma & Mitamura (2025)	Flat text of claim and cited document	Single LLM reasoning pass at inference time	Novelty evaluation
Knappich et al. (2026)	List of features extracted by an LLM	LLM workflow per feature, aggregated to the claim	Feature retrieval, novelty evaluation
MemGraph (Xiong et al., 2025)	Entities with ontology labels recalled from parametric memory	LLM prompted with the recalled memory	Patent matching
Schmitt et al. (2023)	Manually built semantic feature combinations	Mathematical-logical comparison	Patentability assessment
Gao et al. (2024)	Dependency graph over the claims of an application	Trained graph model	Approval prediction
PAGE (this work)	Typed feature graph extracted once per text and cached	Trained graph encoder with edge-aware attention and task heads	Novelty evaluation, prior-art retrieval, patent classification

A.3 Explicit structure and graph neural networks

Gao et al. (2024) show that a fine-grained claim dependency graph predicts patent approval better than much larger language models, and GraphEval (Feng et al., 2025) uses the same division of labour as PAGE outside the patent domain: an LLM decomposes an idea into a graph of viewpoint nodes, and a lightweight graph model produces the evaluation more robustly and far more cheaply than the LLM itself. Among graph encoders, R-GCN (Schlichtkrull et al., 2018) gives every relation type its own weight matrix, so edges of the same type are indistinguishable, whereas the TransformerConv layer (Shi et al., 2021) adds an edge feature to both the attention score and the message, which is what lets PAGE condition attention on the wording of a relation rather than only its type. The attentive readout follows gated graph sequence networks (Li et al., 2016), and all graph layers are implemented in PyTorch Geometric (Fey & Lenssen, 2019).

A.4 Retrieval and sentence representations

BM25 (Robertson & Zaragoza, 2009) remains a strong lexical baseline whenever queries and documents share vocabulary, while dense passage retrieval (Karpukhin et al., 2020) encodes queries and passages into a shared space so that documents are indexed once and searched by inner product; the underlying InfoNCE objective (van den Oord et al., 2018) benefits from large effective batches, which gradient caching (Gao et al., 2021) makes affordable on a single device. Sentence-BERT (Reimers & Gurevych, 2019) established the bi-encoder architecture used here, and MiniLM (Wang et al., 2020) distils it into the compact frozen encoder behind the node and edge features of PAGE.

B Models Used in the Experiments

This appendix describes the models that appear in Section 4 and Section 5, both those that are compared with PAGE and those that PAGE is built from, so that the reported numbers can be read together with the properties of the underlying systems.

B.1 Open-weight LLMs

Five open-weight LLMs are used in two roles: as direct judges that emit the X/A verdict, and as the extractor $\mathcal{E}_{\text{LLM}}$ that converts a text into a feature graph. All of them are decoder-only transformers that are used as released, without any fine-tuning, and the same prompt is used for every model in a given role. Table 6 lists their scale and training budget.

Table 6: The five open-weight LLMs used as judges and as feature-graph extractors. Context denotes the context length supported by the released checkpoint and tokens the size of the pre-training corpus reported by the developers.

Model	Developer	Parameters	Context	Tokens
Qwen2.5-7B (Qwen Team, 2025)	Alibaba	7B	128K	18T
Llama-3.1-8B (Grattafiori et al., 2024)	Meta	8B	128K	15T
Gemma-2-9B (Gemma Team, 2024)	Google DeepMind	9B	8K	8T
Phi-4 (Abdin et al., 2024)	Microsoft	14B	16K	9.8T
Qwen2.5-14B (Qwen Team, 2025)	Alibaba	14B	128K	18T

Qwen2.5-7B and Qwen2.5-14B. The Qwen2.5 series (Qwen Team, 2025) is pre-trained on 18 trillion tokens, up from 7 trillion in Qwen2, and post-trained with supervised fine-tuning on more than one million examples followed by multi-stage reinforcement learning. The architecture combines grouped-query attention, SwiGLU activations, rotary positional embeddings and RMSNorm, and the released checkpoints support a context of 128K tokens with a generation length of 8K tokens. The series is reported to have been improved specifically on the understanding of structured input and on the generation of structured output, in particular JSON, which is the output format that the extraction prompt of PAGE requires.

Llama-3.1-8B. The Llama 3 herd (Grattafiori et al., 2024) is pre-trained on about 15 trillion multilingual tokens and released at 8B, 70B and 405B parameters with a context window of up to 128K tokens. The 8B model is a dense transformer with grouped-query attention and a 128K-entry vocabulary, and it is the smallest model of the herd, which makes it a natural test of whether feature extraction is within reach of a compact model.

Gemma-2-9B. Gemma 2 (Gemma Team, 2024) applies two modifications to the decoder-only transformer that matter at this scale: local sliding-window attention with a window of 4096 tokens is interleaved with global attention over the full context of 8192 tokens, and logits are soft-capped for stability. The 9B model is trained on 8 trillion tokens of web text, code and mathematics with knowledge distillation from a larger teacher instead of plain next-token prediction. Its context of 8192 tokens is the shortest among the five models, which is not a limitation here because the extractor reads one claim or one passage at a time.

Phi-4. Phi-4 (Abdin et al., 2024) is a 14B dense decoder-only transformer trained on 9.8 trillion tokens in which synthetic data generated by multi-agent and self-revision workflows plays a central role, alongside filtered web data, code, academic material and question–answer data. Pre-training uses a context of 4K tokens that is extended to 16K in a mid-training stage, and post-training combines supervised fine-tuning with direct preference optimisation. The model is built for reasoning-heavy tasks at a small scale, which makes the comparison in Table 1 informative: it is the strongest of the five models in the judge role, yet the verdicts it emits remain far below what the same model achieves as an extractor.

B.2 Text encoders and retrieval baselines

all-MiniLM-L6-v2. The frozen sentence encoder ϕ of PAGE is all-MiniLM-L6-v2, a six-layer MiniLM (Wang et al., 2020) of about 22M parameters that maps a text to a 384-dimensional vector and truncates input beyond 256 word pieces. It is fine-tuned with a contrastive objective on more than one billion sentence pairs, and its short truncation limit is inconsequential for PAGE, whose inputs to the encoder are the names of features and the phrases that express relations rather than whole passages. The same checkpoint appears in three further roles: as the zero-shot cosine baseline, as the embedding space of the logistic-regression baseline on $[|\mathbf{a}-\mathbf{b}|; \mathbf{a}\odot\mathbf{b}]$, and, fine-tuned with a linear head, as the supervised text baseline for patent classification. Comparing PAGE with these baselines therefore isolates the contribution of the graph from that of the embeddings, since the embeddings are identical.

BERT cross-encoder. The supervised pair baseline encodes the claim and the passage jointly with BERT (Devlin et al., 2019) and reads the decision from the pooled representation, so that every token of the claim can attend to every token of the passage. This is the strongest form of interaction available to a text model, and it is also the reason why a cross-encoder cannot be used for retrieval over a corpus: the joint encoding has to be recomputed for every candidate, whereas a dual encoder indexes the corpus once.

BM25. BM25 (Robertson & Zaragoza, 2009) scores a passage by the frequencies of the query terms it contains, with saturation in the term frequency and a correction for document length. It is the lexical reference point of the retrieval experiment and indicates how much of the ranking can be recovered from vocabulary overlap alone.

PatentSBERTa. PatentSBERTa (Bekamiri et al., 2024) is an SBERT bi-encoder adapted to patent claims that produces 768-dimensional embeddings (§A.2). It is used in two configurations. The zero-shot configuration measures what a domain-adapted encoder provides without training on this benchmark. The fine-tuned configuration is the controlled comparison for PAGE: it is trained with the same symmetric InfoNCE objective, the same A-category hard negatives, the same effective batch size of 64 realised with gradient caching (Gao et al., 2021), the same epoch budget and the same model-selection rule, so that the representation being ranked is the only difference between the two systems.

B.3 Components of the graph encoder and the ablated variants

Each row of Table 2 replaces exactly one component of PAGE while the extractor, the cached graphs, the data partition and the training budget stay fixed.

TransformerConv against R-GCN. The encoder of PAGE stacks TransformerConv layers (Shi et al., 2021), which add a projection of the edge feature to the key and to the message, so that both the attention weight and the information that is passed depend on the full edge feature, that is on the relation type and on the embedded relation phrase. R-GCN (Schlichtkrull et al., 2018) instead assigns one weight matrix to each relation type. It therefore separates the four relations of the schema but treats all edges of a type as identical, and it is the ablation that isolates the contribution of the relation phrase.

Attentive readout against mean pooling. The attentive readout (Li et al., 2016) scores every node with a small network and returns the weighted sum of node states, which allows a few nodes to dominate the graph embedding. Mean pooling is the variant in which all nodes receive the same weight, and the comparison of the two measures how much the decision depends on concentrating the representation on the distinguishing features.

Interaction head against cosine similarity. The interaction head consumes the ordered concatenation $[\mathbf{h}_a; \mathbf{h}_b; |\mathbf{h}_a - \mathbf{h}_b|; \mathbf{h}_a \odot \mathbf{h}_b]$ and can therefore represent an asymmetric relation between the two graphs. Cosine similarity is symmetric in its two arguments and can only express how close the two embeddings are, which is the reason why this ablation costs the most.

Optimisation and implementation. All trained components are optimised with AdamW (Loshchilov & Hutter, 2019), and the graph layers are those of PyTorch Geometric (Fey & Lenssen, 2019).

C Algorithms

This appendix summarises PAGE as pseudocode. Algorithm 1 covers the offline extraction stage, Algorithm 2 the construction of node and edge features and the shared graph encoder f_θ, and Algorithms 3–5 the training and inference procedures of the three tasks. The encoder architecture is the same for all three tasks, whereas its parameters θ are initialised and trained separately for each task together with the task head g_ψ.

Extraction and caching. Algorithm 1 is run once per extractor LLM. Every unique text is keyed by a hash, and a text whose graph is already cached is skipped, so that texts shared by many pairs are read only once. The output is validated rather than trusted: dangling edges are dropped, and a relation label outside the schema is mapped to INTERACTS_WITH, the most general of the four relations. A text whose output cannot be parsed is represented by a single node carrying the whole text (Algorithm 2).

Algorithm 1 Technical feature graph extraction and caching (Stage 1, offline)

Require: set of patent texts $\mathcal{T}$ (claims and prior-art passages or documents); extractor LLM $\mathcal{E}_{\text{LLM}}$; system prompt P_{sys}; user template $P_{\text{usr}}(\cdot)$ (Appendix D); relation set $\mathcal{R}$
Ensure: graph cache $\mathcal{K}$ mapping $\text{hash}(t)$ to $G_t = (V_t, E_t)$

for each unique text $t \in \mathcal{T}$ **do**
  $k \leftarrow \text{hash}(t)$
  if $k \in \mathcal{K}$ **then continue** ▷ each text is extracted once
  end if
  $o \leftarrow \mathcal{E}_{\text{LLM}}\big(P_{\text{sys}}, P_{\text{usr}}(t)\big)$ ▷ greedy decoding
  $J \leftarrow$ PARSEFIRSTJSON(o) ▷ strip code fences, match braces
  if $J = \varnothing$ **or** J.nodes $= \varnothing$ **or** some node lacks `id` or `text` **then**
    continue ▷ not cached; single-node fallback in Alg. 2
  end if
  $V \leftarrow J$.nodes; $I \leftarrow \{v.\texttt{id} : v \in V\}$; $E \leftarrow \varnothing$
  for each edge $e \in J$.edges **do**
    if $e.\texttt{source} \notin I$ **or** $e.\texttt{target} \notin I$ **then continue** ▷ drop dangling edges
    end if
    if $e.\texttt{relation} \notin \mathcal{R}$ **then** $e.\texttt{relation} \leftarrow$ INTERACTS_WITH
    end if
    if $e.\texttt{text}$ is missing **then** $e.\texttt{text} \leftarrow e.\texttt{relation}$
    end if
    $E \leftarrow E \cup \{e\}$
  end for
  $\mathcal{K}[k] \leftarrow (t, V, E)$
end for
return $\mathcal{K}$

Graph encoder. Algorithm 2 turns a cached graph into tensors and encodes it. The sentence encoder ϕ is frozen, so the feature matrices $\mathbf{X}$ and $\mathbf{E}$ of every text are computed once and reused in every epoch; only the TransformerConv layers and the readout are trained.

Algorithm 2 Node and edge features and the graph encoder f_θ (Stage 2)

function BUILDGRAPH$(t, \mathcal{K}, \phi)$ ▷ computed once per text
  if $\text{hash}(t) \in \mathcal{K}$ **then** $(V, E) \leftarrow \mathcal{K}[\text{hash}(t)]$
  else $V \leftarrow \{v_1\}$ with $\text{txt}(v_1) = t$; $E \leftarrow \varnothing$ ▷ single-node fallback
  end if
  for each $v_i \in V$ **do** $\mathbf{x}_i \leftarrow \phi(\text{txt}(v_i)) \in \mathbb{R}^{384}$
  end for
  for each $e_{ij} \in E$ **do** $\mathbf{e}_{ij} \leftarrow [\,\text{onehot}(r_{ij});\ \phi(\text{txt}(e_{ij}))\,] \in \mathbb{R}^{388}$
  end for
  return $G = (\mathbf{X}, \mathbf{E}, E)$
end function

function ENCODE$_\theta(G)$
  $\mathbf{H} \leftarrow \text{ReLU}\big(\text{LN}\big(\text{TConv}^{(1)}_{\text{4 heads}}(\mathbf{X}, E, \mathbf{E})\big)\big)$ ▷ edge-aware attention, Eq. (3)–(4)
  $\mathbf{H} \leftarrow \text{Dropout}(\mathbf{H})$
  $\mathbf{H} \leftarrow \text{ReLU}\big(\text{LN}\big(\text{TConv}^{(2)}_{\text{1 head}}(\mathbf{H}, E, \mathbf{E})\big)\big)$
  for each node $i \in V$ **do** $s_i \leftarrow \mathbf{w}^\top \mathbf{h}_i + b$
  end for
  $\beta_i \leftarrow \exp(s_i) \,/\, \sum_{j \in V} \exp(s_j)$ ▷ softmax within the graph
  return $\mathbf{h}_G \leftarrow \sum_{i \in V} \beta_i \mathbf{h}_i$
end function

Novelty evaluation. Algorithm 3 trains the Siamese model on labelled claim–passage pairs. The partition by connected components of the bipartite claim–passage graph guarantees that no claim and no passage occurs in more than one split. Model selection uses the threshold-free validation AUC, and the decision threshold is fixed on the validation split before the test split is evaluated.

Algorithm 3 Training and inference for novelty evaluation

Require: labelled pairs $\mathcal{D} = \{(a, b, y)\}$ with $y = 1$ for X and $y = 0$ for A; graphs G_a, G_b from Alg. 2

$\mathcal{D}_{\text{tr}}, \mathcal{D}_{\text{va}}, \mathcal{D}_{\text{te}} \leftarrow$ split $\mathcal{D}$ by connected components of the claim–passage graph
$w_+ \leftarrow |\{y = 0\}| \,/\, |\{y = 1\}|$ in $\mathcal{D}_{\text{tr}}$; initialise θ, ψ
for epoch $= 1, \ldots, T_{\max}$ **do**
  for each mini-batch $\mathcal{B} \subset \mathcal{D}_{\text{tr}}$ **do**
    $\mathbf{h}_a \leftarrow \textsc{Encode}_\theta(G_a),\ \mathbf{h}_b \leftarrow \textsc{Encode}_\theta(G_b)$ ▷ shared weights
    $z \leftarrow \text{MLP}_\psi\big([\mathbf{h}_a;\, \mathbf{h}_b;\, |\mathbf{h}_a - \mathbf{h}_b|;\, \mathbf{h}_a \odot \mathbf{h}_b]\big)$
    $\mathcal{L} \leftarrow -\frac{1}{|\mathcal{B}|} \sum_{\mathcal{B}} \big[w_+\, y \log \sigma(z) + (1 - y) \log\big(1 - \sigma(z)\big)\big]$
    update θ, ψ with AdamW and gradient-norm clipping
  end for
  compute AUC on $\mathcal{D}_{\text{va}}$; reduce learning rate on plateau
  keep (θ, ψ) with the best validation AUC; stop early after P epochs without gain
end for
$\tau^\star \leftarrow \arg\max_\tau \text{F1}\big(\mathcal{D}_{\text{va}};\, \sigma(z) \geq \tau\big)$ ▷ threshold fixed on validation data
return test predictions $\hat{y} = \mathbb{1}[\sigma(z) \geq \tau^\star]$ on $\mathcal{D}_{\text{te}}$

Prior-art retrieval. Algorithm 4 trains the dual encoder. Each query claim contributes one positive pair per X-category passage, and for each query in a mini-batch one A-category passage cited against the same claim is sampled as a hard negative that is shared by all queries of the batch. After training, the corpus is encoded once, and each query costs one graph encoding and one inner product per document.

Algorithm 4 Training and inference for prior-art retrieval

Require: training queries $\mathcal{Q}_{\text{tr}}$; X-category passages $R(q)$ and A-category passages $A(q)$ of each query; corpus $\mathcal{C}$; scale $s = 20$

$\mathcal{P} \leftarrow \{(q, p) : q \in \mathcal{Q}_{\text{tr}},\, p \in R(q)\}$; initialise θ
for epoch $= 1, \ldots, T_{\max}$ **do**
  for each mini-batch $\{(q_k, p_k)\}_{k=1}^{B}$ of shuffled $\mathcal{P}$ **do**
    $\bar{\mathbf{h}}_{q_k} \leftarrow \text{norm}(\textsc{Encode}_\theta(G_{q_k})),\ \bar{\mathbf{h}}_{p_k} \leftarrow \text{norm}(\textsc{Encode}_\theta(G_{p_k}))$
    $N_B \leftarrow \{\, n_k \sim \text{Uniform}(A(q_k)) : A(q_k) \neq \varnothing \,\}$ ▷ shared hard negatives
    $S_{kc} \leftarrow s\, \langle \bar{\mathbf{h}}_{q_k}, \bar{\mathbf{h}}_c \rangle$ for $c \in \{p_1, \ldots, p_B\} \cup N_B$
    $S'_{kl} \leftarrow s\, \langle \bar{\mathbf{h}}_{p_k}, \bar{\mathbf{h}}_{q_l} \rangle$ for $l = 1, \ldots, B$ ▷ passage-to-query term
    $\mathcal{L} \leftarrow \frac{1}{2}\big[\text{CE}(S, \text{diag}) + \text{CE}(S', \text{diag})\big]$; update θ with AdamW
  end for
  rank the validation corpus for every validation query; compute MAP
  keep θ with the best validation MAP; stop early after P epochs without gain
end for
$\mathcal{I} \leftarrow \{\bar{\mathbf{h}}_d : d \in \mathcal{C}\}$ ▷ index built once
for each test query q **do**
  rank $d \in \mathcal{C}$ by $\langle \bar{\mathbf{h}}_q, \bar{\mathbf{h}}_d \rangle$
end for

Patent classification. Algorithm 5 trains the single-graph classifier. Model selection uses the validation macro-F1 score, which weights every section equally, consistent with the evaluation criterion in Section 5.4.

Algorithm 5 Training and inference for patent classification

Require: documents $\mathcal{D} = \{(d, c)\}$ with IPC section $c \in \{1, \dots, C\}$; graphs G_d from Alg. 2
1: $\mathcal{D}_{\text{tr}}, \mathcal{D}_{\text{va}}, \mathcal{D}_{\text{te}} \leftarrow$ class-stratified $70/15/15$ split of $\mathcal{D}$
2: $w_c \leftarrow n \,/\, (C\, n_c)$ from the class counts n_c of $\mathcal{D}_{\text{tr}}$; initialise θ, ψ
3: **for** epoch $= 1, \dots, T_{\max}$ **do**
4: **for** each mini-batch $\mathcal{B} \subset \mathcal{D}_{\text{tr}}$ **do**
5: $\boldsymbol{\ell}_d \leftarrow g_\psi\big(\textsc{Encode}_\theta(G_d)\big) \in \mathbb{R}^C$
6: $\mathcal{L} \leftarrow -\sum_{\mathcal{B}} w_{c_d} \log \mathrm{softmax}(\boldsymbol{\ell}_d)_{c_d} \,/\, \sum_{\mathcal{B}} w_{c_d}$
7: update θ, ψ with AdamW and gradient-norm clipping
8: **end for**
9: compute macro-F1 on $\mathcal{D}_{\text{va}}$; reduce learning rate on plateau
10: keep (θ, ψ) with the best validation macro-F1; stop early after P epochs without gain
11: **end for**
12: **return** $\hat{c}_d = \arg\max_c \boldsymbol{\ell}_{d,c}$ for $d \in \mathcal{D}_{\text{te}}$

D PROMPTS

D.1 TECHNICAL FEATURE GRAPH EXTRACTION

The same system and user prompts are used for claims and for prior-art passages.

System prompt.

```
You are a patent technical expert and graph modeling specialist. You will be given a SHORT
     piece of patent text. It may be either (a) a single (often dependent) patent claim
     fragment, or (b) a passage from a cited patent's description. Decompose it into a
     technical feature graph.

Output must be strict JSON with exactly two keys: "nodes" and "edges". No prose, no Markdown,
     no code fences.

# What to model
Model the substantive TECHNICAL CONTENT only: components, functional units, steps, materials,
     signals, and their parameters/properties. The graph must work equally well for a claim
     fragment and for a descriptive passage - do NOT assume a "Claim 1 system" hierarchy and
     do NOT assume a full set of claims is present.

# Ignore boilerplate cross-references
Inputs like "The X according to claim 6", "Method according to claim 1 wherein", "according to
      any one of claims 6 to 8" are cross-references whose target text is NOT provided. Do NOT
      create nodes for "claim 6" or "claims 6 to 8". Skip the reference preamble and model
     only the technical entities and the substantive limitation that follows (typically after
     "wherein"). You may still create a node for the main subject mentioned (e.g., "electronic
      circuit") if it participates in a relation.

# Node Rules
Extract each technical entity, functional unit, step, material, signal, or parameter as a node
     .
Attributes (exactly these three, no "claims" field):
- id: unique identifier (e.g., "T1", "T2").
- subject: 3-5 word summary of the entity (e.g., "Coupling circuit", "Extrusion temperature").
- text: the original phrase describing this node, copied from the input.

# Edge Rules
Capture relationships between nodes.
Attributes:
- source: source node id.
- target: target node id.
- relation: must be exactly one of ["CONSISTS_OF", "INTERACTS_WITH", "IS_A", "HAS_PROPERTY"].
   - CONSISTS_OF: structural / part-of (e.g., Device -> Component, Method -> Step, "comprising
       ", "includes", "further comprising").
   - INTERACTS_WITH: dynamic functional flow, data/energy transfer, connection, or logical
       trigger (e.g., "connects ... to", "sends ... to", "based on", "when operating in").
   - IS_A: taxonomy / type / instance (e.g., "the sensor is a LiDar sensor", "is shown as", "
       is a type of").
   - HAS_PROPERTY: an entity has a parameter, attribute, value, material, or qualitative
       property (e.g., "having a resistivity value", "at a temperature of 200 to 260C", "made
        of polyolefin", "relatively invariant to changes under load"). Use this for the
       parametric/qualitative limitations that often decide novelty.
- text: [CRITICAL] copy the original verb/preposition/predicate from the input (e.g., "
     connects ... to", "may include", "is shown as", "having", "at a temperature of").

# Guidance
```

```
1. Keep it faithful: every node/edge must be grounded in the input text. Do not invent
     components that are not mentioned.
2. Short inputs may yield small graphs (2-5 nodes). That is fine - do not pad.
3. Prefer HAS_PROPERTY for numeric ranges, material choices, and qualitative descriptors.
4. Ensure the JSON is valid and strictly follows this schema.
```

User prompt.

```
Decompose the following patent text into a technical feature graph. Follow the system rules
     strictly. Output strict JSON only.

Input Text:

<<INPUT_TEXT>>

Reminders:
- Three node attributes only: id, subject, text. Do NOT add a "claims" field.
- Ignore cross-reference boilerplate ("according to claim N", "according to any one of claims
     X to Y"); model only the substantive technical content.
- relation must be one of: CONSISTS_OF, INTERACTS_WITH, IS_A, HAS_PROPERTY.
- In each edge's "text", copy the original predicate/preposition from the input.

Few-shot Examples:

Example 1 - a dependent claim fragment with a cross-reference:
Input:
The electronic circuit according to any one of claims 6 to 8 wherein when it operates in the
     respective second operating mode the first coupling circuit connects the first additional
      capacitor to a reference potential.
Output:
{
"nodes": [
{"id": "T1", "subject": "Electronic circuit", "text": "The electronic circuit"},
{"id": "T2", "subject": "First coupling circuit", "text": "the first coupling circuit"},
{"id": "T3", "subject": "First additional capacitor", "text": "the first additional capacitor
     "},
{"id": "T4", "subject": "Reference potential", "text": "a reference potential"},
{"id": "T5", "subject": "Second operating mode", "text": "the respective second operating mode
     "}
],
"edges": [
{"source": "T1", "target": "T2", "relation": "CONSISTS_OF", "text": "the first coupling
     circuit"},
{"source": "T2", "target": "T3", "relation": "INTERACTS_WITH", "text": "connects ... to"},
{"source": "T3", "target": "T4", "relation": "INTERACTS_WITH", "text": "to a reference
     potential"},
{"source": "T2", "target": "T5", "relation": "INTERACTS_WITH", "text": "when it operates in"}
]
}

Example 2 - a passage from a cited description, with properties:
Input:
The dampening resistive element is shown as a fixed value resistor which is relatively
     invariant to changes in resistance under load. In some embodiments the dampening
     resistive element may include multiple fixed value resistors each having a different
     resistivity value.
Output:
{
"nodes": [
{"id": "T1", "subject": "Dampening resistive element", "text": "The dampening resistive
     element"},
{"id": "T2", "subject": "Fixed value resistor", "text": "a fixed value resistor"},
{"id": "T3", "subject": "Invariant under load", "text": "relatively invariant to changes in
     resistance under load"},
{"id": "T4", "subject": "Multiple resistors", "text": "multiple fixed value resistors"},
{"id": "T5", "subject": "Resistivity value", "text": "a different resistivity value"}
],
"edges": [
{"source": "T1", "target": "T2", "relation": "IS_A", "text": "is shown as"},
{"source": "T2", "target": "T3", "relation": "HAS_PROPERTY", "text": "which is relatively
     invariant"},
{"source": "T1", "target": "T4", "relation": "CONSISTS_OF", "text": "may include"},
{"source": "T4", "target": "T5", "relation": "HAS_PROPERTY", "text": "each having"}
]
}

Use this structure as the strict output format guideline.
```

Example output. The graph in Figure 2 is returned in the following form.

```
{"nodes": [{"id": "T1", "subject": "Working vehicle", "text": "The working vehicle 1"},
        {"id": "T2", "subject": "Fuel tank", "text": "a fuel tank 50"},
        {"id": "T3", "subject": "Exhaust gas purification device",
         "text": "the exhaust gas purification device 21"},
        {"id": "T4", "subject": "Engine", "text": "an engine E"},
        {"id": "T5", "subject": "Heat shield member", "text": "a heat shield member 51"}],
 "edges": [{"source": "T1", "target": "T2", "relation": "CONSISTS_OF", "text": "comprising"},
        {"source": "T2", "target": "T3", "relation": "HAS_PROPERTY",
         "text": "is disposed in back of"},
        {"source": "T2", "target": "T4", "relation": "INTERACTS_WITH",
         "text": "supplies a fuel to"},
        {"source": "T1", "target": "T5", "relation": "CONSISTS_OF", "text": "comprising"},
        {"source": "T5", "target": "T3", "relation": "INTERACTS_WITH", "text": "separates"},
        {"source": "T5", "target": "T2", "relation": "INTERACTS_WITH", "text": "separates"}]}
```

D.2 DIRECT LLM JUDGEMENT

```
[SYSTEM]
You are a patent examiner assessing NOVELTY. Given a patent CLAIM and a PRIOR-ART PASSAGE,
     decide whether the passage discloses the claimed invention and would DESTROY the novelty
     of the claim (this corresponds to an X-type citation). If the passage is only topically
     related but does NOT by itself destroy novelty (an A-type citation) or is unrelated, it
     does NOT conflict. Answer with exactly one word: Yes or No.

[USER]
CLAIM: The blade according to claim 8 or 9 wherein the purge valley is configured having at
     least one of a maximum depth axially at a position axially upstream of the leading edge
     and a maximum depth laterally at a location forward the leading edge of the airfoil.
PASSAGE: There has been a demand to further broaden the scope of the technology disclosed in
     US Patent Publication No.6419446 for the purpose of providing the turbine blade cascade
     capable of reducing the secondary flow loss irrespective of the fluctuation in the flow
     rate of the working fluid and discord between the edge line of the rounded cusp-like
     protruding portion and the stagnation point of the working fluid.
Answer Yes or No.

[ASSISTANT]
Yes

[USER]
CLAIM: The wireless communications device of claim 12 wherein the remote destination is a
     preconfiguration server configured to provide preconfiguration information for one or
     more subsequent short-range communication sessions with the proximate wireless
     communications device.
PASSAGE: Then mutual normal authentication based upon the init keys created at both sides is
     performed. The PBU inquires of the storage device of a Bluetooth identifier which
     identifies the Bluetooth transceiver circuitry of the mobile terminal from which the
     message originated. Both the storage device and the link are secured, preventing
     unauthorized access to the stored information.
Answer Yes or No.

[ASSISTANT]
No

[USER]
CLAIM: <<CLAIM>>
PASSAGE: <<PRIOR_ART_PASSAGE>>
Does the passage destroy the novelty of the claim? Answer Yes or No.

[ASSISTANT]
```