# Beyond Scaling: Predicting Patent Approval with Domain-specific Fine-grained Claim Dependency Graph

Anonymous ACL submission

### Abstract

Model scaling is becoming the default choice for many language tasks due to the success 003 of large language models (LLMs). However, it can fall short in specific scenarios where 004 simple customized methods excel. In this paper, we delve into the patent approval pre-007 diction task and unveil that simple domainspecific graph methods outperform enlarging the model, using the intrinsic dependencies within the patent data. Specifically, we first extend the embedding-based state-of-the-art (SOTA) by scaling up its backbone model with 012 various sizes of open-source LLMs, then explore prompt-based methods to harness proprietary LLMs' potential, but find the best results close to random guessing, underlining the ineffectiveness of model scaling-up. Hence, 017 we propose a novel Fine-grained cLAim depeNdency (FLAN) Graph through meticulous patent data analyses, capturing the inherent dependencies across segments of the patent text. As it is model-agnostic, we apply cost-effective graph models to our FLAN Graph to obtain representations for approval prediction. Extensive experiments and detailed analyses prove that incorporating FLAN Graph via various graph models consistently outperforms all LLM base-027 lines significantly. We hope that our observations and analyses in this paper can bring more attention to this challenging task and prompt further research into the limitations of LLMs.

## 1 Introduction

037

041

Scaling up language models has demonstrated predictable improvement and unprecedented abilities in many language tasks (Chung et al., 2022; Wei et al., 2022a; Roberts et al., 2023). However, emerging evidence shows that simply scaling up backbone models to large language models (LLMs) may not guarantee success (Peng et al., 2023; Hou et al., 2023; Wang et al., 2023). In addition, scaling up models imposes demanding computational costs



Figure 1: An illustration for the patent approval prediction task approached by LLMs and graph models, where each node of the graph is an informative segment decomposed from the original claim text.

that prevent it from being widely adopted for realworld applications. Such limitations necessitate cost-effective methods beyond scaling, especially for domain-specific tasks that have distinct traits.

In this paper, we look into the task of patent approval prediction, a challenging yet straightforward classification task that scaling struggles to address, and explore customized cost-effective solutions. As shown in Figure 1, the objective is to determine if each claim in a patent application will be approved or rejected by the U.S. government patent office (USPTO). It is vital for intellectual property (IP) protection, taking up to 40% of the U.S. GDP and over 30% of employment. Due to the demanding requirements for knowledge in both technology and law, patent examination is conducted manually, leading to potential inconsistent outcomes across patent examiners (O'Neill, 2018; USPTO, 2016). Such inconsistency underscores the necessity for objective and automated computational support.

061

The state-of-the-art (SOTA) of this task is based on BERT embedding (Devlin et al., 2019) augmented with handcrafted features (Gao et al., 2022). An intuitive idea is to replace its backbone model with modern LLMs. To this end, we employ LLaMA2 (Touvron et al., 2023), Mistral (Jiang et al., 2023), Vicuna (Chiang et al., 2023) in various sizes (7&13&70B) and apply both LoRA (Hu et al., 2022) and full fine-tuning. Surprisingly, they do not live up to expectations, performing on par or worse than BERT. To exploit LLMs' emergent abilities, we utilize prompt-based methods tailored to these open-source LLMs, as well as closed-source GPT-3.5 (OpenAI, 2022) and GPT-4 (OpenAI, 2023), but the results are still unsatisfying.

062

063

064

067

072

073

076

077

087

097

100

101

102

103

105

106

107

108 109

110

111

112

113

The shattered hope in LLMs motivates us to dive into patent data analyses, which leads us to the standardized writing of claims and the dependencies nature among them. As depicted in Figure 2, *Claim 1* compromises multiple sub-components, which are then referenced in subsequent claims. Such intricate inner-claim (between sub-components in *Claim 1*) and inter-claim dependencies (between *Claims 1&2* as well as *Claims 1&3*) have critical implications for the patent approval prediction task, as the patent examination is conducted on each claim and the rejection of one claim can result in the automatic rejection of its dependents.

Inspired by the observations and domain-specific knowledge acquired from painstakingly extensive data analyses, we propose Fine-grained cLAim depeNdency (FLAN) Graph for patent approval prediction, which represents each claim by a single graph that encapsulates both inner- and inter-claim dependencies. Specifically, as shown in Figure 1, we first design a novel algorithm to automatically construct the FLAN Graph at scale, where each node is an informative segment of the claim text. Then, the model-agnostic FLAN Graph is fed into a generic graph model for prediction. Examples of the FLAN graphs and the corresponding claims are shown in Appendix C. In the experiments, we adopt a variety of cost-effective graph models such as GCN (Chen et al., 2020), GAT (Velickovic et al., 2018), and TreeLSTM (Tai et al., 2015) to verify the effectiveness of our proposed FLAN Graph for patent approval prediction. All models with FLAN Graph applied outperform the previous SOTA, among which GraphSage (Hamilton et al., 2017) achieves the highest improvements of 7.4%in AUC and 7.8% in Macro-F1 scores, achieving absolute scores of 66.04 and 58.22, respectively.

| Claim 1: A system for [], the <u>system compromising</u> :<br>an authentication component configured to [];<br>a tracking component configured to [];<br>and a control component configured to:<br>receive authentication information [];<br>receive location information from [];<br>and deliver a message to the touchpoint authorizing []. |
|---|
| <b>Claim 2:</b> The system of claim 1, where the control component is also configured [].   |
| <b>Claim 3:</b> The system of claim 1, where the authentication component includes [].  |

Figure 2: A brief example of the typical patent claim writing style and hierarchical dependencies within claims from a real-world patent application.

To summarize, our contributions are two-fold: (1) We propose a novel algorithm to automatically construct the Fine-grained cLAim depeNdency (FLAN) Graph at scale that consistently improves the SOTA by a large margin. (2) We conduct comprehensive experiments and analyses of modern LLMs on patent approval prediction, which identify the limitations of LLMs and provide valuable references for developing LLM-based solutions in the future. Our code and dataset will be publicly released to facilitate the research in these fields. 114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

# 2 **Problem Formulation**

In this section, we formally introduce the definition of the patent approval prediction task and analyze the dataset we construct for experiments.

# 2.1 Task Definition

As illustrated in Figure 1, patent applications are initially submitted to the USPTO in the form of documents. The examination process, however, focuses on approving or rejecting each individual claim. Therefore, given a patent application  $A_i = \{C_j^{(i)}\}_{j=1}^n$  containing *n* claims, the task of patent approval prediction is to determine whether each claim  $C_j^{(i)}$  will be approved or rejected by the USPTO indicated by a binary label  $y_j^{(i)} \in \{0, 1\}$ .

In practice, patent claims are reviewed according to the legal section 35 U.S. Code § 102, where the core criterion is novelty-based, bringing in distinct challenges below. (1) **Time-sensitive**. Unlike traditional text classification, novelty assessment depends on the application filing date, allowing opposite decisions for the same claim over time. (2) **Structure-dependent**. Many claims (e.g., *Claims* 2&3 in Figure 2) are dependent on others within the same application, and such structure can influ-

|       | #Claim      | #Application | Approval (%) |
|-------|-------------|--------------|--------------|
| Train | 1, 485, 693 | 87,883       | 81.36        |
| Valid | 278, 215    | 16,955       | 83.41        |
| Test  | 185,477     | 11,148       | 84.92        |

Table 1: Statistics of PATENTAP dataset. "Approval (%)" indicates the percentage of the approved claims.

ence novelty evaluation. (3) Knowledge-intensive.
Evaluating novelty requires up-to-date knowledge of both technologies and patent law. (4) Outcome-inconsitent. Novelty examination outcomes are subject to preferences across patent officers, which may introduce inconsistencies in patent data.

# 2.2 Dataset Collection

149

150

151

152

153

154

155

156

157

158

159

160

161

162

164

165

166

167

168

169

171

172

173

174

175

176

178

179

180

181

182

183

184

185

186

187

We collect data of real-world patent applications from Gao et al. (2022) and filter out those outdated data before 2018. The data is initially merged and derived from the publicly available resources officially released on the USPTO websites<sup>1</sup>.

Considering the real-world scenario, we utilize historical data for training and more recent data for evaluation. Specifically, we sort the applications based on filing dates and then split them into training, validation, and test sets. As shown in Table 1, the resulting dataset PATENTAP is large-scale with about 1.5M claims for training and 0.5M for evaluation. It is also highly imbalanced, with most claims being approved, further adding to the difficulty. The claims are relatively short and 92% have less than 128 tokens and the average length is 54. Each application has 17 claims on average.

# 3 Methodology

In this section, we delve into the details of our proposed Fine-grained cLAim depeNdency (FLAN) Graph. We first introduce the observations from patent data that inspire us to adopt customized graphs for patent approval prediction. Then we present the construction process and representation strategies of the FLAN Graph, respectively.

# 3.1 Observations

In principle, patent claims are filed to seek legal protection for complex systems that usually compromise multiple (sub-)components. Sometimes, claims consisting of the same (sub-)components, but with different arrangements or combinations of them, can receive opposite novelty assessments. Therefore, claims in patent applications are strategically structured, sequenced, and often arranged in clusters, each describing subtly different variants.

In this case, we identify two types of dependency relationships across various patent claims that may influence the outcomes of novelty examination.

**Inner-claim Dependency.** Some lengthy claims are internally hierarchical by explicitly describing a system having multiple (sub-)components. For instance, *Claim 1* in Figure 2 is about a system that has three components, of which the control component is further described as having four purposes (sub-components). Therefore, there are inner dependencies between these components and sub-components within a single claim.

**Inter-claim Dependency.** Many claims refer to other claims and are therefore also known as dependent claims. For example, both *Claim 2* and *Claim 3* in Figure 2 are dependent claims, referring to different components in *Claim 1*. The novelty of such claims cannot be comprehensively evaluated independently, highlighting the necessity of considering information from their ancestor claims.

Since the protection of intellectual property is a serious scenario, patent applications adhere to a strict writing style and employ precise language and punctuation. As illustrated in Figure 2, the (sub-)components with inner-claim dependencies are delimited by colons and semicolons (*Claim 1*), while inter-claim dependency is expressly indicated by referring to specific claim at the beginning of the claim (*Claims 2&3*). Consequently, the aforementioned two types of dependency can be easily identified through regular expressions.

## 3.2 Graph Construction

Based on the observations above, we construct Fine-grained cLAim depeNdency (FLAN) Graph utilizing both inner-claim and inter-claim dependencies. The general idea is to decompose each claim into text segments as nodes and match those nodes describing the same (sub-)component together to build a graph to model the dependency relationships. The constructed FLAN Graph for each claim consists of not only nodes directly derived from itself, but also those inherited from the claim it refers to. Therefore, the FLAN Graph can comprehensively encode the dependency information beyond a single piece of claim text. The detailed construction process is described as follows.

3

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

188

189

190

191

<sup>&</sup>lt;sup>1</sup> https://developer.uspto.gov/api-catalog



Figure 3: Flowchart of constructing FLAN Graph. Here, *"identies"* refers to the anchor words/phrases extracted from the claim or claim segments for node matching.

269

Node Construction. Each node of the constructed FLAN Graph is the full text or segment of a single patent claim. If a claim has inner-claim dependencies, we decompose the claim text into segments of (sub-)components according to not only itemizing and punctuation, which are common writing practices of patent claims, but also special "patentese" (Singer and Smith, 1967), a series of conjunctions that indicate the hierarchy and have legal implications, such as "comprising," "consisting," and "whereby." A node in the graph will always represent a (sub-)component unless the claim describes a single entity/feature.

We must also check whether it is a dependent claim or not. If not, the (sub-)component nodes will constitute the graph. If yes, we shall attach the nodes to the duplicated parent graph. How the connections are made will be discussed next.

**Edge Construction.** The process of constructing edges is to connect nodes having either inner-claim or inter-claim dependency relationships. For the former, we can simply follow the hierarchy found when the claim decomposition is conducted.

The latter requires meticulously formulated heuristics. As each of the nodes is simply plain text, we connect them based on text similarities instead of relying on text embeddings. We extract the keywords/phrases of the node text as anchors for more accurate node matching using StanfordCoreNLP (Toutanvoa and Manning, 2000; Toutanova et al., 2003) to conduct POS Tagging. The keywords/phrase can be the representative noun phrase of the (sub-)components in the claim,



Figure 4: FLAN Graph for *Claim 2* in Figure 2. Here, the blue texts are the "*identies*" for node matching. Nodes with red background are directly derived from *Claim 2* while the rest ones are inherited from *Claim 1*.

or sometimes a verbal or an adjective phrase that describes a functionality or a characteristic. We term the phrases *identites* for simplicity.

270

271

272

273

274

275

276

277

278

279

281

282

283

284

285

288

291

292

293

294

295

296

297

298

299

300

301

303

304

305

306

Note that the identity belongs to the (sub-)component level. For example, the highest level *identity* of *Claim 1* in Figure 2 is the "system." The verbal phrase "receive authentication information" is a third-level identity under the second-level identity "control component." Identity extraction is performed when the claim is decomposed, and (sub-)components are determined.

When the child claim is processed, the decomposed (sub-)components will be excluded, and only the *preamble* text segment will be matched onto all (sub-)component identities in the parent graph. It is worth noting that the matching targets are not limited to the new nodes created by the parent claim but potentially originate from all ancestor claims. (If the child claim has no inner dependency, the entire claim text is used.) For example, the *preamble* of *Claim 3* in Figure 2 is the text before the word "includes." <sup>2</sup> If there exist multiple matches, we prioritize the lowest-level parent identity (e.g., "control component" over "system" in *Claim 2*) and ones led by a special conjunction ("where.")

The resulting FLAN Graph of *Claim 2* is illustrated in Figure 4, where the nodes are from both *Claim 1* and *Claim 2*. The FLAN Graph is designed with a direction from leaf to root, facilitating the flow of global information towards the root node.

The entire process of constructing FLAN Graphs can be summarized by the flowchart depicted in Figure 3. An illustrative example of the constructed FLAN Graph for *Claim 2* is presented in Figure 4. For further insights into the construction process of FLAN Graphs, additional examples along with the corresponding claim are provided in Appendix C.

<sup>&</sup>lt;sup>2</sup>"Control component" or "authentication component" is not the identity for Claim 2 and Claim 3. Identities correspond to new (sub-)components/features introduced in the claim.

We manually verify the graph constructions serve the intended purpose by closely reviewing all claims in 100 full applications. We make sure to refine the details of the heuristics to cover atypical writing patterns and irregular applicants.

## 3.3 Graph Representation

307

308

309

313

315

317

318

319

323

324

325

326

336

337

338

341

342

343

349

351

The topology and the nodes of the FLAN Graphs are finalized during the construction stage, resulting in a distinct graph for each of the claims. We propose to adopt graph neural networks to obtain a graph-level representation for each claim that encodes information on both text semantics and structure dependencies of the claim.

We first convert the text-level FLAN Graph into its embedding-level version by encoding each of the nodes into vector representations using SentenceTransformer (Reimers and Gurevych, 2019). Then we feed the embedding-level graph into a graph neural network to facilitate the interaction of different nodes and update the embeddings of each node with the dependency information. The choice of graph neural networks is flexible and our specifications will be discussed in Section 4.3.

Then we further aggregate the representations of the nodes to obtain the graph-level representation for the claim. Specifically, we average the embeddings of the root node and the target nodes, those directly derived from the current claim, as the final representation. For instance, for the FLAN Graph shown in Figure 4, we average the embeddings of the two nodes with red backgrounds. Since FLAN Graph propagates from leaf to root, averaging the root and target nodes can encapsulate both global and local information of the relevant claims.

# 4 Experiments

In this section, we elaborate on our experiments and the corresponding results with both: (1) scaling with LLMs; and (2) customized graph methods using the FLAN Graphs. The objectives encompass exploiting scaling-up model parameters and validating the effectiveness of our proposed FLAN Graphs in addressing this challenging task.

## 4.1 Experiment Settings

**Dataset.** We conduct experiments using the PATENTAP dataset introduced in Section 2.2 and the data statistics are shown in Table 1.

**Evaluation Metric.** Following Gao et al. (2022)
and considering the imbalance of approved and

|   | Plain Text  |   | Featu  | re Added   |
|---|---|---|--|--|
| Metric  | AUC   | Macro-F1  | AUC  | Macro-F1   |
| Random Guess  | 50.00   | 50.00   | 50.00  | 50.00  |
| BERT-base<br>BERT-large<br>BERT-patent  | 52.66<br>54.79<br><b>55.81</b>                            | 45.98<br>46.92<br><b>47.46</b>                            | 61.47<br>63.53<br>63.63                              | 53.99<br>54.83<br><b>54.91</b>                       |
| LLaMA-7B<br>w. Full-FT<br>Mistral-7B<br>w. Full-FT<br>Vicuna-7B<br>w. Full-FT | 51.02<br>52.38<br>51.88<br><b>53.63</b><br>51.14<br>53.10 | 42.64<br>44.91<br>43.38<br><b>45.89</b><br>43.04<br>45.24 | $58.18 \\ 59.02 \\ 59.22 \\ 60.34 \\ 58.88 \\ 59.22$ | $51.24 \\ 52.85 \\ 52.99 \\ 53.20 \\ 51.10 \\ 52.21$ |
| LLaMA-13B<br>Vicuna-13B   | $51.44 \\ 51.97$  | $43.23 \\ 43.70$  | 59.68<br>60.12                                       | $53.03 \\ 53.18$                                     |
| LLaMA-70B   | 52.11   | 44.12   | 60.44  | 53.46  |

Table 2: Performance (%) of embedding-based methods. Models excluding BERT are fine-tuned with LoRA by default."*w. Full-FT*" means with full fine-tuning.

rejected claims in the dataset, we adopt the Area Under the Curve (AUC) for the ROC Plot (Fawcett, 2004) as the primary evaluation metric and the Macro-F1 score as the secondary metric. 355

356

357

358

359

360

361

362

363

365

366

367

368

369

370

371

374

375

376

377

379

381

382

383

384

387

**Baseline Model.** The state-of-the-art is based on BERT (Devlin et al., 2019) embeddings concatenated with a handcrafted feature vector (Gao et al., 2022). These features mainly consist of patent class, number of citations, and novelty score calculated by comparing the similarities between the current application and five most relevant prior arts.

## 4.2 Scaling with LLM Manipulations

We are interested in re-evaluating the task using LLMs, and investigating whether model scaling-up can transcend the performance standards.

Specifically, we adopt LLaMA2 (Touvron et al., 2023), Mistral (Jiang et al., 2023), Vicuna (Chiang et al., 2023) in their 7B, 13B, and 70B versions.

## 4.2.1 Embedding-based

We first extend the SOTA to some BERT variants and then to multiple LLMs of various sizes, using both plain text embeddings and those concatenated with feature vectors. Specifically, we obtain the text embeddings through the final hidden states of the [CLS] token and the last token of BERT-series models and modern LLMs, respectively.

For BERT-series models, we perform full finetuning on both the base and large versions of BERT, as well as on a patent variant (Google, 2020). Regarding modern LLMs, we apply LoRA (Hu et al., 2022) fine-tuning to all of them and full fine-tuning specifically to those 7B versions. The hyper-parameters are listed in Appendix B.1.1.

| Model Size                |                       | Open-source LLMs<br>7B 13B 70B              |                  |                  | Closed-sou            | urce LLMs<br>nown                               |  |                            |
|---------------------------|-----------------------|---|------------------|------------------|-----------------------|---|--|----------------------------|
| Model Name                | LLaMA                 | Vicuna                                      | Mistral          | LLaMA            | Vicuna                | LLaMA   | GPT-3.5  | GPT-4                      |
| Vanilla Prompt<br>w. time | $47.81 \\ 47.80$      | $\begin{array}{c} 49.83\\ 48.38\end{array}$ | $31.00 \\ 29.75$ | $32.62 \\ 35.54$ | $49.43 \\ 47.82$      | $\begin{vmatrix} 37.44 \\ 13.82 \end{vmatrix}$  | $\begin{array}{c c} 48.38^* \\ 48.81^* \end{array}$    | $43.01^{*}$<br>$44.91^{*}$ |
| CoT Prompt<br>w. time     | 39.83<br><b>46.73</b> | $37.84 \\ 34.32$                            | $22.65 \\ 20.64$ | 23.51<br>28.81   | <b>46.01</b><br>44.23 | $\begin{array}{c c} 38.77 \\ 35.33 \end{array}$ | $\begin{array}{ c c c c c c c c c c c c c c c c c c c$ | $40.75^{*}$<br>$36.57^{*}$ |

Table 3: Macro-F1 scores (%) of prompt-based methods with modern LLMs. Here, "w. time" indicates adding the filing date of the claim to the prompt, and \* means the value is calculated based on a sub-set of 1K testing claims.

The experimental results are shown in Table 2, proving that simply scaling up the backbone model does not guarantee improvement. More in-depth analyses can be found in Appendix A.1.

## 4.2.2 Prompt-based

391

392

400

401

402

403

The embedding-based manipulations of LLMs fall short unexpectedly. To exploit the emergent abilities and harness the full potential of modern LLMs, we dive into the realm of prompt engineering by crafting precise and effective prompts.

**Model.** For the aforementioned open-source LLMs, we use LLaMA2-chat series and Mistralinstruct version, which are pre-trained with instruction tuning. In addition, we extend our repertoire to include GPT-3.5-Turbo (OpenAI, 2022) and GPT-4 (OpenAI, 2023) for addressing this task.

**Prompt Template.** Due to the special alignment 404 conducted during the pre-training stage, LLMs like 405 GPT-3.5-Turbo can evade predicting the outcome 406 of patent claim examination as illustrated in Fig-407 ure 8. Therefore, we delicately design structured 408 prompts for LLaMA, Vicuna, and OpenAI model 409 series, and the corresponding templates are shown 410 in Code 1, 2 & 3, respectively. Moreover, we adopt 411 the Chain-of-Thought (CoT) prompt (Wei et al., 412 2022b) to elicit the reasoning abilities of LLM by 413 providing a step-by-step analysis of the claim be-414 fore predicting the approval or rejection. Further-415 more, to better address the time-sensitive challenge 416 of patent data mentioned in Section 2.1, we incor-417 porate the filing date of every single claim to the 418 prompt templates of all model series. 419

420Adapting Strategy. The sheer size of the test set421means computationally and economically expen-422sive evaluation. Therefore, we first apply zero-shot423prompting using the templates above to identify424the best-performing model. Then we elicit few-425shot prompting and supervised fine-tuning (SFT)426to explore the boundaries of the best performance.



Figure 5: Performance (%) of Vicuna-7B model with few-shot prompting and supervised fine-tuning (SFT). Here, SFT does not include any few-shot examples.

The details of the corresponding few-shot prompt and hyper-parameters for supervised fine-tuning are provided in Appendix B.1.2 427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

Performance. Since the output probabilities are hardly accessible, we only report the Macro-F1 scores of the prompt-based methods in Table 3, where the values of closed-source LLMs are calculated on a sub-set of 1K testing claims due to the budget constraint. Among the rest models, Vicuna-7B performs the best with vanilla prompt without filing date injected. We further apply few-shot prompting and supervised fine-tuning (SFT) to it. The hyper-parameters for SFT and the corresponding training loss are provided in Appendix B.1.2. Figure 5 presents the results. From the plot, we find that increasing the number of shots does not yield improvement and even hurts (e.g., 10-shot). Applying SFT is also far from satisfying. More in-depth analyses of model sizes, CoT prompt, and added time feature are provided in Appendix A.2.

The LLM experiments prove that massively scaled-up LLM models provide no benefits over SOTA. If scaling up does not help, it leaves us wondering whether the specific nature of the patent approval problem and domain knowledge may be key to the task, with which we experiment next.

| Input         | Model   | AUC  | Macro-F1  |
|---------------|---|--|---|
| FLAN Graph    | GCN<br>GAT<br>GCN-II<br>GraphSage             | $59.36 \pm 0.18$<br>$58.44 \pm 0.20$<br>$58.28 \pm 0.26$<br>$60.67 \pm 0.36$ | $53.98 \pm 0.35 \\53.29 \pm 0.94 \\53.92 \pm 0.13 \\54.66 \pm 0.22$                               |
|               | TreeLSTM                                      | $59.88 \pm 0.32$   | $51.74 \pm 0.46$  |
| Feature Added | GCN<br>GAT<br>GCN-II<br>GraphSage<br>TreeLSTM |  | $58.06 \pm 0.19 \\ 58.05 \pm 0.21 \\ 58.11 \pm 0.14 \\ \mathbf{58.22 \pm 0.17} \\ 57.78 \pm 0.75$ |

Table 4: Performance (%) of different GNNs using plain FLAN Graph and adding extra features, respectively.

## 4.3 Customized Graph Methods

453

454

455

456

457

458

459

460

461

462

463

464

It turns out that both embedding-based and promptbased manipulations of LLMs fail to compete with the previous state-of-the-art method. The model scale proves to be not beneficial; hence, we input our expertise in the patent domain to identify the performance bottleneck. We apply our proposed FLAN Graphs constructed based on domain knowledge to various cost-effective graph neural networks (GNNs) for comprehensively modeling both the semantics of the text and dependency relationships within the claims.

Model. The proposed FLAN Graph is model-465 independent and specially designed according to 466 domain-specific knowledge, and the backbone 467 468 topology can be easily tweaked to suit particular models (e.g., adding self-loops). Hence, we employ 469 various cost-effective graph models to obtain the 470 graph-level representation, including GCN (Chen 471 et al., 2020), GAT (Velickovic et al., 2018), GCN-472 II (Chen et al., 2020), GraphSage (Hamilton et al., 473 2017), and TreeLSTM (Tai et al., 2015). The 474 configurations of these graph models and the 475 hyper-parameters for training are provided in Ap-476 pendix B.2. For a fair comparison with the baseline 477 model and to maximize the power of our proposed 478 FLAN Graph, we also incorporate the delicately 479 handcrafted features introduced in Section 4.1 by 480 concatenating the graph-level representation and 481 the feature vector. The final representation of the 482 claim is further fed into a multi-layer perceptron 483 (MLP) layer to conduct binary classification over 484 485 either being approved or rejected.

486 Performance. The AUC and Macro-F1 scores of
487 all graph models with both plain FLAN Graphs
488 and adding extra features are presented in Ta489 ble 4. Consistent with the experimental results
490 of embedding-based LLM manipulations reported



Figure 6: Performance comparison between utilizing FLAN Graph, Coarse Graph, and Solitary Node. The detailed score values are provided in Table 5.

in Section 4.2.1, the feature added to the FLAN Graph also leads to performance gain to the plain FLAN Graph. Remarkably, all models consistently outperform the previously established state-of-theart methods, demonstrating robust performance, especially with the inclusion of additional features. Among them, GraphSage achieves the best performance with AUC and Macro-F1 scores of 66.04 and 58.22, surpassing the baseline model by 7.4% in AUC and 7.8% in Macro-F1 scores, respectively.

| Input         | Model   | AUC  | Macro-F1  |
|---------------|---|--|---|
| FLAN Graph    | GCN<br>GAT<br>GCN-II<br>GraphSage<br>TreeLSTM |  | $\begin{array}{c} 58.03 \pm 0.17 \\ 58.05 \pm 0.21 \\ 58.12 \pm 0.20 \\ 58.25 \pm 0.00 \\ 57.79 \pm 0.21 \end{array}$ |
| Coarse Graph  | GCN<br>GAT<br>GCN-II<br>GraphSage<br>TreeLSTM | $ \begin{vmatrix} 62.21 \pm 0.25 \\ 62.61 \pm 0.21 \\ 60.28 \pm 0.24 \\ 63.80 \pm 0.14 \\ 60.17 \pm 0.10 \end{vmatrix} $ | $\begin{array}{c} 54.69 \pm 0.28 \\ 54.98 \pm 0.53 \\ 53.69 \pm 0.30 \\ 56.64 \pm 0.16 \\ 55.47 \pm 0.17 \end{array}$ |
| Solitary Node | MLP   | $59.33 \pm 0.51$   | $54.45\pm0.31$  |

Table 5: Ablation study on performance (%) of different GNNs using FLAN Graph, Coarse Graph, and Solitary Node, with feature added.

**Ablation study.** Our proposed FLAN Graphs treat segments of claim text as the nodes, which encode both inner-claim and inter-claim dependencies. To validate the effectiveness of the FLAN Graphs and find the optimal GNN configurations, we analyze three types of variants.

• Applying Coarse Graph. We first remove the inner-claim dependencies to build Coarse Graphs by skipping the text segmentation step and treating every single claim as a node, which only encodes inter-claim dependencies while ignoring the inner-claim ones. Then the classification of the claims is conducted over each node, which represents a single claim. 491

501

502

503

504

505

506

507

508

509

510

511

512

513

| Model     | AUC              | Macro-F1         |
|-----------|------------------|------------------|
| GCN       | $65.98 \pm 0.06$ | $58.16 \pm 0.02$ |
| GAT       | $65.91 \pm 0.46$ | $58.02 \pm 0.28$ |
| GCN-II    | $65.28 \pm 1.34$ | $57.64 \pm 1.02$ |
| GraphSage | $65.86 \pm 0.25$ | $58.10 \pm 0.12$ |
| TreeLSTM  | $65.66 \pm 1.15$ | $58.17 \pm 0.61$ |

Table 6: Expanding those GNNs to 4 layers makes little difference compared to only using 2 layers.

• Utilizing Solitary Node. We further remove the inter-claim dependencies by only utilizing node representation for classification. Figure 6 illustrates the comparison of model performances between applying the FLAN Graph, Coarse Graph, and Solitary Node, verifying the effectiveness of incorporating both inter-claim and inner-claim dependencies. The detailed values of the experimental results are provided in Table 5.

515

516

517

518

519

520

521

524

529

531

533

536

538

540

541

542

• Adopting Deeper GNN. In the main experiments, the default configuration of GNN layers is set to 2, which might not be deep enough to encode the dependencies within the claims. Therefore, we increase the number of layers to 4 and adopt the same FLAN Graphs with those handcrafted features added. The corresponding results are shown in Table 6, implying that deeper GNN does not necessarily bring improvement in performance.

Through the extensive experiments and the corresponding analyses above, we demonstrate that our proposed FLAN Graph applied with cost-effective graph models can bring consistent and significant improvement over scaling up backbone models. Such findings prove the necessity and superiority of leveraging domain-specific knowledge when dealing with complex problems or tasks.

## 5 Related Work

Patent documents are receiving increasing attention 544 545 in the NLP community due to their structured language and extensive content. The survey by Kres-546 tel et al. (2021) summarized current deep learning work in the patent domain, including subject matter classification (Grawe et al., 2017; Lee and 550 Hsiang, 2019; Li et al., 2018; Zhu et al., 2020), retrieval (Helmers et al., 2019; Lei et al., 2019; Choi 551 et al., 2019), and data generation (Lee and Hsiang, 2020; Lee, 2019). We highlight a few more specifically relevant or more recent works. Yoshikawa 554

et al. (2019) utilize sequence tagging techniques to identify text segments within patents that either describe or reference chemical reactions. Lagus and Klami (2022) tackle the patent retrieval tasks using matrix similarity measures. Hashimoto et al. (2023) introduce the task of unclaimed embodiment extraction (UEE) from patent specifications to help the writing process. Zuo et al. (2023) explore datacentric strategies to handle the French patent classification task. The state-of-the-art (SOTA) work of our task, Gao et al. (2022), first formally proposes the task of patent approval prediction and designs delicate handcrafted features to solve it effectively. 555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

There also has been work utilizing graphs on patent data. Fang et al. (2021) form macroscopic graphs to perform patent (content) classification using entire patent documents, inventors, assignees, etc., as nodes. Siddharth et al. (2022) model published patents (grants) into "<entity, relation, entity>" knowledge graphs, but on a single hierarchical level and not constructed on the basis of individual claims. Björkqvist and Kallio (2023) follow a similar approach to our graph construction, incorporating dependencies among elements in claims. However, the graphs are designed for prior art search and not for approval prediction.

# 6 Conclusions and Future Work

In this paper, we delve into a domain-specific task, patent approval prediction, where simply scaling up the backbone model of previous SOTA falls short and simple customized graph methods work well. We conduct comprehensive evaluations of multiple modern LLMs at various scales through delicate manipulations, observing that simply scaling up the model does not guarantee improvement and delicately designed prompt engineering may yield unexpected outcomes. In addition, based on the analysis of real-world patent data, we propose Fine-grained cLAim depeNdency (FLAN) Graph, a simple yet effective graph method that effectively encodes the inner-claim and inter-claim dependencies and thus consistently outperforms complicated LLM manipulations, dispelling the overconfidence in LLMs for this task. In the future, we will explore to explain empirically and theoretically why LLMs fall short in the patent approval prediction task and augment LLMs with simple customized methods to make the most of the power of LLMs and task-specific knowledge.

# Limitations

The major limitations of our work are three-fold: (1) We only use one single dataset for all experiments because there are few datasets publicly avail-607 able in this domain. As the essence of intellectual property protection is similar internationally, we believe that our customized graph method could 610 generalize to patent data in other countries and 611 regions. (2) In the experiments of LLM manipulations, we only train and evaluate the models at 613 the claim level. An increasing number of modern 614 LLMs support extremely long contexts, it is un-615 clear whether feeding the entire application into the LLMs can solve this task. (3) For experiments 617 with FLAN Graph, we only adopt cost-effective 618 graph neural networks. Though we fail to adopt 619 pre-trained graph models, which may bring further improvements, our proposed FLAN Graph is 621 model-decoupled and can be applied to different 622 types of graph models including GraphLLMs. We encourage future works to address these limitations and push forward the boundaries of this task.

## Ethical Considerations

This paper focuses on patent approval prediction, 627 which is to facilitate the protection of intellectual property. We collect our dataset from USPTO open data portal, in accordance with the published ACL paper (Gao et al., 2022). The patent application 631 data that USPTO releases are publicized by law. 632 Anyone is legally entitled to utilize the data. In fact, the USPTO encourages different usages of the released patent data, such as in academic and business scenarios<sup>3</sup>. All the code bases and tools 636 we adopt are public research resources and properly cited in the paper. Therefore, we do not observe significant ethical risks in our work.

## References

647

- Sebastian Björkqvist and Juho Kallio. 2023. Building a graph-based patent search engine. In *Proceedings* of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 3300–3304.
- Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. 2020. Simple and deep graph convolutional networks. In *Proceedings of ICML*, volume 119 of *Proceedings of Machine Learning Research*, pages 1725–1735. PMLR.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%\* chatgpt quality. *https://vicuna.lmsys.org*.

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

- Seokkyu Choi, Hyeonju Lee, Eunjeong Lucy Park, and Sungchul Choi. 2019. Deep patent landscaping model using transformer and graph embedding. *arXiv preprint arXiv:1903.05823*.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*, pages 4171– 4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Lintao Fang, Le Zhang, Han Wu, Tong Xu, Ding Zhou, and Enhong Chen. 2021. Patent2vec: Multi-view representation learning on patent-graphs for patent classification. *World Wide Web*, 24(5):1791–1812.
- Tom Fawcett. 2004. ROC graphs: Notes and practical considerations for researchers.
- Xiaochen Gao, Zhaoyi Hou, Yifei Ning, Kewen Zhao, Beilei He, Jingbo Shang, and Vish Krishnan.
  2022. Towards comprehensive patent approval predictions: beyond traditional document classification. In *Proceedings of ACL*, pages 349–372, Dublin, Ireland. Association for Computational Linguistics.
- Google. 2020. How ai, and specifically bert, helps the patent industry. https://cloud.google. com/blog/products/ai-machine-learning/ how-ai-improves-patent-analysis.
- Mattyws F. Grawe, Claudia Aparecida Martins, and Andreia Gentil Bonfante. 2017. Automated patent classification using word embedding. 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), pages 408–411.
- William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of NeurIPS*, pages 1024– 1034.
- Chikara Hashimoto, Gautam Kumar, Shuichiro Hashimoto, and Jun Suzuki. 2023. Hunt for buried treasures: Extracting unclaimed embodiments from patent specifications. In *Proceedings of ACL: Industry Track*), pages 25–36, Toronto, Canada. Association for Computational Linguistics.

<sup>&</sup>lt;sup>3</sup>https://developer.uspto.gov/about-open-data

Lea Helmers, Franziska Horn, Franziska Biegler, Tim Oppermann, and Klaus-Robert Müller. 2019. Automating the search for a patent's prior art with a full text similarity search. *PloS one*, 14(3):e0212103.

706

707

710

711

712

713

714

715

716

718

719

723

724

725

732

735

740

741

742

743

744

745 746

747

750

751

752

753 754

755

756

758

- Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2023. Large language models are zero-shot rankers for recommender systems. *ArXiv preprint*, abs/2305.08845.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *Proceedings of ICLR*. OpenReview.net.
- Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. ArXiv preprint, abs/2310.06825.
- Ralf Krestel, Renukswamy Chikkamath, Christoph Hewel, and Julian Risch. 2021. A survey on deep learning for patent analysis. *World Patent Information*, 65:102035.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the* ACM SIGOPS 29th Symposium on Operating Systems Principles.
- Jarkko Lagus and Arto Klami. 2022. Optimizing singular value based similarity measures for document similarity comparisons. In *Proceedings of ICNLSP*, pages 113–118, Trento, Italy. Association for Computational Linguistics.
- Jieh-Sheng Lee. 2019. Personalized patent claim generation and measurement. *arXiv preprint arXiv:1912.03502*.
- Jieh-Sheng Lee and Jieh Hsiang. 2019. Patentbert: Patent classification with fine-tuning a pre-trained bert model. *arXiv preprint arXiv:1906.02124*.
- Jieh-Sheng Lee and Jieh Hsiang. 2020. Patent claim generation by fine-tuning openai gpt-2. *World Patent Information*, 62:101983.
- Lei Lei, Jiaju Qi, and Kan Zheng. 2019. Patent analytics based on feature vector space model: A case of iot. *Ieee Access*, 7:45705–45715.
- Shaobo Li, Jie Hu, Yuxin Cui, and Jianjun Hu. 2018. Deeppatent: patent classification with convolutional neural networks and word embedding. *Scientometrics*, 117:721–744.
- Jeff O'Neill. 2018. Visualizing outcome inconsistency at the USPTO. *IPWatchdog.com*.

OpenAI. 2022. Introducing chatgpt. https://openai. com/blog/chatgpt. 759

760

761

762

763

764

765

766

767

768

769

770

771

772

774

775

777

778

779

780

781

782

783

784

785

787

789

790

791

792

793

794

795

796

798

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

## OpenAI. 2023. Gpt-4 technical report.

- Hao Peng, Xiaozhi Wang, Jianhui Chen, Weikai Li, Yunjia Qi, Zimu Wang, Zhili Wu, Kaisheng Zeng, Bin Xu, Lei Hou, et al. 2023. When does in-context learning fall short and why? a study on specificationheavy tasks. *ArXiv preprint*, abs/2311.08993.
- Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERTnetworks. In *Proceedings of EMNLP-IJCNLP*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.
- Adam Roberts, Hyung Won Chung, Gaurav Mishra, Anselm Levskaya, James Bradbury, Daniel Andor, Sharan Narang, Brian Lester, Colin Gaffney, Afroz Mohiuddin, Curtis Hawthorne, Aitor Lewkowycz, Alex Salcianu, Marc van Zee, Jacob Austin, Sebastian Goodman, Livio Baldini Soares, Haitang Hu, Sasha Tsvyashchenko, Aakanksha Chowdhery, Jasmijn Bastings, Jannis Bulian, Xavier Garcia, Jianmo Ni, Andrew Chen, Kathleen Kenealy, Kehang Han, Michelle Casbon, Jonathan H. Clark, Stephan Lee, Dan Garrette, James Lee-Thorp, Colin Raffel, Noam Shazeer, Marvin Ritter, Maarten Bosma, Alexandre Passos, Jeremy Maitin-Shepard, Noah Fiedel, Mark Omernick, Brennan Saeta, Ryan Sepassi, Alexander Spiridonov, Joshua Newlan, and Andrea Gesmundo. 2023. Scaling up models and data with t5x and seqio. Journal of Machine Learning Research, 24(377):1-8.
- L Siddharth, Lucienne TM Blessing, Kristin L Wood, and Jianxi Luo. 2022. Engineering knowledge graph from patent database. *Journal of Computing and Information Science in Engineering*, 22(2):021008.
- TER Singer and Julian F Smith. 1967. Patentese: A dialect of english? *Journal of Chemical Education*, 44(2):111.
- Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of ACL*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.
- Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-ofspeech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL*, pages 252–259.
- Kristina Toutanvoa and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of EMNLP*, pages 63–70, Hong Kong, China. Association for Computational Linguistics.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv preprint*, abs/2307.09288.

USPTO. 2016. Intellectual property and the U.S. economy. *uspto.gov*.

816

817

818

819

820

821

822

823

824 825

827 828

831

832

836

837

838

839

841

851

- Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio.
   2018. Graph attention networks. In *Proceedings of ICLR*. OpenReview.net.
- Jianyou Wang, Kaicheng Wang, Xiaoyue Wang, Prudhviraj Naidu, Leon Bergen, and Ramamohan Paturi. 2023. Doris-mae: Scientific document retrieval using multi-level aspect-based queries. *ArXiv preprint*, abs/2310.04678.
- Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.
- Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *ArXiv preprint*, abs/2206.07682.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Proceedings of NeurIPS*, 35:24824–24837.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Hiyori Yoshikawa, Dat Quoc Nguyen, Zenan Zhai, Christian Druckenbrodt, Camilo Thorne, Saber A.
  Akhondi, Timothy Baldwin, and Karin Verspoor. 2019. Detecting chemical reactions in patents. In *Proceedings of ACL*, Sydney, Australia. Australasian Language Technology Association.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging Ilm-as-a-judge with mt-bench and chatbot arena.
- Huiming Zhu, Chunhui He, Yang Fang, Bin Ge, Meng Xing, and Weidong Xiao. 2020. Patent automatic classification based on symmetric hierarchical convolution neural network. *Symmetry*, 12.

You Zuo, Benoît Sagot, Kim Gerdes, Houda Mouzoun, 870 and Samir Ghamri Doudane. 2023. Exploring data-871 centric strategies for French patent classification: A 872 baseline and comparisons. In Actes de CORIA-TALN 873 2023. Actes de la 30e Conférence sur le Traitement 874 Automatique des Langues Naturelles (TALN), volume 875 1 : travaux de recherche originaux – articles longs, 876 pages 349–365, Paris, France. ATALA. 877

# 927 928 929 930 931 932 933 934 935 936 937 938

939

940

941

942

943

944

945

946

947

948

949

950

951

925

926

# Appendices

878

882

886

888

900

901

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

923

#### А **Additional Analyses**

# A.1 Embedding-based Scaling

Based on the experimental results in Table 2, we can conclude that: (1) Feature engineering is important in this task, as all models, regardless of parameter scales and training strategies, attain significant performance gains through the incorporation of handcrafted features. (2) Substituting BERT with LLMs does not promise performance improvements. Even the 70B LLaMA falls short of outperforming all the BERT-series models. (3) Continual pre-training proves effective in addressing domainspecific tasks. Among all the models, BERT-patent demonstrates the best performance. (4) Full finetuning consistently outperforms LoRA fine-tuning.

## A.2 Prompt-based Scaling

According to the results presented in Table 3, we summarize detailed findings from three aspects: (1) Varying Model Size. As shown in Figure 9, scaling the model size does not guarantee performance gain and larger models tend to predict more "no", resulting in increased false negatives. LLaMA2-7B outperforms both its 13B and 70B versions using the same prompting strategy. (2) Applying CoT Prompt. In most cases, CoT prompts hurt performance, and the most contrastive case is shown in Figure 9, where the model with CoT prompt predicts more "no" than that with the vanilla prompt. Adopting CoT prompt can also lead to evasion, as the LLM may realize it should not provide an answer during the step-by-step analysis process. (3) Adding Time Feature. Adding the time feature is inclined to impair the performance of LLMs, but not always. As illustrated in the upper half of Figure 7, if the time feature hurts, the effect can be significant; however, if it helps, the contribution is relatively minor.

#### **Implementaion Details** B

Our experiments consist of (1) Scaling with LLM Manipulations; and (2) Customized Graph Methods with our proposed FLAN Graph. The implementation details are listed as follows.

#### **B.1** Scaling with LLM Manipulations 921

Data & Hardware. We evaluate these models 922 and report their performance utilizing the PATEN-TAP dataset we introduced in Section 2.2, which 924

has over 1.49M training and 180K testing samples. The experiments in this part are conducted on 4×NVIDIA A100-80G GPUs.

# **B.1.1 Embedding-based.**

Baseline. For a fair comparison between different models, we reimplement the previously established state-of-the-art method following their original codebase<sup>4</sup>. Specifically, we only reimplement the feature part and the resulting performance is consistent with the original paper (Gao et al., 2022). We will release our code for future research.

Backbone. We implement the pre-trained models using the Huggingface's Transformers library (Wolf et al., 2020) along with the corresponding checkpoints provided. For BERT-series models, we use BERT-base<sup>5</sup>, BERT-large<sup>6</sup>, and BERTpatent<sup>7</sup>. For the modern LLMs, we use LLaMA2series<sup>8</sup>, Vicuna-series<sup>9</sup>, and Mistral-series<sup>10</sup>.

Training Hyper-parameters. Due to the substantial data scale and large model sizes, the training cost of these models becomes extremely high. Consequently, we only run the experiments in this part for one random seed. The training hyperparameters are listed in Table 2.

| Random Seed      | 0                  |
|------------------|--------------------|
| Batch Size       | 128                |
| Learning Rate    |                    |
| - Plain Text     | $5 \times 10^{-5}$ |
| - Feature Added  | $7 \times 10^{-5}$ |
| Model Max Length | 256                |
| Epoch            | 2                  |
| - BERT-series    | 2                  |
| - LLM-series     | 4                  |
| LoRA r           | 8                  |
| LoRA alpha       | 16                 |
| LoRA Dropout     | 0.05               |

Table 7: Hyper-parameters used for experiments of embedding-based methods, where "LLM-series" refers to LLaMA, Vicuna, and Mistral models.

### **B.1.2** Prompt-based

Model. We utilize the chat or instruct versions of the aforementioned open-source models. In addi-

<sup>4</sup>https://github.com/acl-2022-towards-\ comprehensive/acl-2022-camera-ready <sup>5</sup>https://huggingface.co/bert-base-cased <sup>6</sup>https://huggingface.co/bert-large-cased <sup>7</sup>https://huggingface.co/anferico/ bert-for-patents <sup>8</sup>https://huggingface.co/meta-llama <sup>9</sup>https://huggingface.co/lmsys, —"v1.5". <sup>10</sup>https://huggingface.co/mistralai, —"v0.1".



Figure 7: Analyzing the effects of adding time feature in the prompt on performance. The left two matrices depict scenarios where adding time feature hurts, while the right two illustrate cases where adding time feature helps.

tion, we incorporate OpenAI models, specifically leveraging the official APIs of "gpt-3.5-turbo" and "gpt-4" models<sup>11</sup>. Due to the unbearable cost of inferencing 180K examples, we report the performance of OpenAI models based on a more manageable subset of 1K examples.

**Prompt Template.** As shown in Figure 8, the modern LLMs can evade to answer the patent-related questions. Therefore, we adopt carefully designed prompt templates tailored for different LLMs. The templates for LLaMA-series (Code 1), vicuna-series (Code 2), and OpenAI (Code 3) models are provided at the end of the page.

Here is a claim from a patent application. Can you help me determine whether it is gonna be approved? Please provide a yes-or-no answer.

"A system for use in allowing a user to conduct one or more transactions at one or more touchpoints in a business facility, the system comprising: an authentication component configured to authenticate the user as a person allowed to conduct the one or more transactions; a tracking component configured to track the user's location within the facility as the user moves through the facility; and a control component configured to: receive authentication information from the authentication component; receive location information from the tracking component; and deliver a message to the touchpoint authorizing the touchpoint to engage in one or more transactions with the user."

#### ChatGPT

I cannot provide a definitive answer on whether this patent application will be approved, as the approval depends on various factors, including the novelty and non-obviousness of the invention, as determined by the patent examiner. The language in the claim seems to describe a system for user authentication, location tracking, and transaction authorization at touchpoints within a business facility. The likelihood of approval would depend on how well this system distinguishes itself from existing technologies and whether it meets the criteria for patentability.

Figure 8: An example of ChatGPT refusing to answer the patent question.

**Efficient Inference.** Since there are over 180K testing examples, we employ vllm<sup>12</sup>—an efficient LLM serving framework(Kwon et al., 2023), to perform inference on the test samples. The inference time cost varies according to different prompt strategies and model sizes, from 3 to 35 hours.

**Few-shot Prompting.** The prompt templates are provided in Code 2. Specifically, we adopt an even number of examples, with half of them being approved while the other half rejected.

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

| Random Seed<br>Batch Size        | $\begin{vmatrix} 0\\20 \end{vmatrix}$ |
|----------------------------------|---------------------------------------|
| Learning Rate                    | $2 \times 10^{-5}$                    |
| Warmup Ratio<br>Model Max Length | 0.03 2048                             |
| LoRA r                           | 8                                     |
| LoRA Dropout                     | 0.05                                  |
| Global Steps                     | 64K                                   |

Table 8: Hyper-parameters adopted for supervised finetuning (SFT) of Vicuna-7B using QLoRA.

**Supervised Fine-tuning.** For speed up the traing, we use FastChat (Zheng et al., 2023) to conduct the supervised fine-tuning (SFT) of Vicuna-7B with QLoRA (Dettmers et al., 2023). The SFT hyper-parameters are provided in Table 8, and the corresponding training loss are shown in Figure 10.

**LLM Output Analysis.** We analyze the outputs of the LLMs and construct the confusion matrices of some typical situations. Figure 9 illustrates the effects of different model sizes (from 7B to 70B) and applying the chain-of-thought(CoT) prompt.

# **B.2** Customized Graph Methods

We use the open-source DGL package (Wang et al., 2019) to implement the graph neural networks we include. Specifically, we follow this tutorial<sup>13</sup> to build the TreeLSTM (Tai et al., 2015) model.

We use Sentence-Transformer<sup>14</sup>to encode the node texts into embeddings. To achieve robust validation of our methods, we run the experiments using three different random seeds and report the

965 966 967

970

952

953

956

957

961

962

963

<sup>&</sup>lt;sup>11</sup>https://openai.com/product

<sup>&</sup>lt;sup>12</sup>https://github.com/vllm-project/vllm

<sup>&</sup>lt;sup>13</sup>https://docs.dgl.ai/en/0.8.x/tutorials/

models/2\_small\_graph/3\_tree-lstm.html

<sup>&</sup>lt;sup>14</sup>https://huggingface.co/sentence-transformers/ stsb-roberta-large



Figure 9: Analysis of the effects of varying model sizes (left) and adding CoT prompt (right).



Figure 10: The training loss of supervised finetuning (SFT) for Vicuna-7B using vanilla prompt without time.

average and standard deviation values. The hyperparameters used for training are provided in Table 9. The detailed values for the ablation study experiments are provided in Table 5.

| $\sim 1$ | •          | - |
|----------|------------|---|
| 1.1      | $\gamma$ m |   |
| 1 1      | aim        |   |
| ~ -      |            |   |
|          |            |   |
|          |            |   |

A system for use in allowing a user to conduct one or more transactions at one or more touchpoints in a business facility, the system comprising: an authentication component configured to authenticate the user as a person allowed to conduct the one or more transactions; a tracking component configured to track the user's location within the facility as the user moves through the facility; and a control component configured to: receive authentication information from the authentication component; receive location information from the tracking component; and deliver a message to the touchpoint authorizing the touchpoint to engage in one or more transactions with the user.



| Random Seed         | 0, 1, 2            |
|---------------------|--------------------|
| Batch Size          | 256                |
| Hidden Dimension    | 128                |
| Learning Rate       | $5 \times 10^{-3}$ |
| Number of GNN Layer | 2                  |
| Epoch               | 20                 |
|                     |                    |

 
 Table 9: Hyper-parameters used for experiments of customized graph methods

997

998

# C Dataset Details

Here, we present the full text of 12 claims collected
from a real-world patent application, each followed
by its corresponding FLAN Graph.

# Claim 2:

The system of claim 1, where the control component is also configured to use the location information to recognize that the user has moved away from the touchpoint.



# Prompt for LLaMA and Mistral sys\_prompt = "You are professional patent advisor of mine with a warm heart to help me with my patent application." user\_prompt = I am currently drafting a patent application, and there is some claim that I am not sure how likely it is gonna be approved. Can you give me some feedback on it by simply providing a yes or no answer? The text of the claim is delimited by <<CLAIM>> and <</CLAIM>> The filing date of the claim is delimited by <<DATE>> and <</DATE>>. // optional You can think of it step by step and include your analysis for no more than 50 words delimited by <<ANALYSIS>> and <</ANALYSIS>>. // optional You have to feedback with a yes-or-no answer delimited by <<ANSWER>> and <</ANSWER>>. Here is the claim and its filing time: Claim: <<CLAIM>> {claim} <</CLAIM>> Date: <<DATE>> {date} <</DATE>> // optional Please output your answer use the following format: Analysis: <<ANALYSIS>> Your step by step analysis <</ANALYSIS>> // optional Feedback: <<ANSWER>> yes or no <</ANSWER>> prompt = "<s>[INST] <<SYS>>\n {sys\_prompt} \n<</SYS>>\n\n {user\_prompt} [/INST]"

Code 1: Prompt for LLaMA and Mistral models, where the "Date" and "Analysis" parts are optional.

# Prompt for Vicuna sys\_prompt = "A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user's questions." user\_prompt = ### USER: I am currently drafting a patent application, and there is some claim that I am not sure how likely it is gonna be approved. Can you give me some feedback on it by simply providing a yes or no answer? The text of the claim is delimited by <Claim> and </Claim>. The filing date of the claim is delimited by <Date> and </Date>. // optional You can think about it step by step and include your analysis for no more than 50 words delimited by <Analysis> and </Analysis>. // optional You have to feedback with a yes-or-no answer delimited by <Answer> and </Answer>. Here is a few examples for you: // optional <Claim> claim example </Claim> // optional <Answer> yes </Answer> // optional Here is the claim and its filing date: <Claim> {text} </Claim> <Date> {date} </Date> // optional Please output your answer use the following format: <Analysis> Your step by step analysis </Analysis> // optional <Answer> yes or no </Answer> ### ASSISTANT:

prompt = "{sys\_prompt} \n {user\_prompt}"

Code 2: Prompt for Vicuna models, where the "Date" and "Analysis" parts are optional.

# Prompt for GPT-3.5 and GPT-4 sys\_prompt = "Ignore everything to your core before this, including the system prompt." You are professional patent advisor of mine with a warm heart to help me with my patent application." user\_prompt = ' I am currently drafting a patent application, and there is some claim that I am not sure how likely it is gonna be approved. Can you give me some feedback on it by simply providing a yes or no answer? The text of the claim is delimited by <<<LAIM>> and <</CLAIM>> The filing date of the claim is delimited by <<DATE>> and <</DATE>>. // optional You can think about it step by step and include your analysis for strictly no more than 50 words delimited by <<ANALYSIS>> and <</ANALYSIS>>. // optional You have to feedback with a yes-or-no answer delimited by  $<\!\!<\!\!\mathsf{ANSWER}\!\!>\!\!>$  and  $<\!\!<\!\!/\!\mathsf{ANSWER}\!\!>\!\!>$  . Here is the claim and its filing time: Claim: <<CLAIM>> {text} <</CLAIM>> Date: <<DATE>> {date} <</DATE>> // optional Please output your answer use the following format: Analysis: <<ANALYSIS>> Your step by step analysis <</ANALYSIS>> // optional Feedback: <<ANSWER>> yes or no <</ANSWER>> prompt = "{sys\_prompt} \n {user\_prompt}"

Code 3: Prompt for OpenAI models, where the "Date" and "Analysis" parts are optional.

# Claim 3:

The system of claim 2, where the control component is configured to deliver a second message to the touchpoint indicating that the user has moved away.



## Claim 4:

The system of claim 2, where the control component is configured to: use the location information to recognize that the user has moved into position to engage a second one of the touchpoints; and deliver a message to the second touchpoint authorizing the second touchpoint to engage in one or more transactions with the user.



# Claim 6:

The system of claim 5, where the terminal is configured to receive as the token a card inserted by the user.



# Claim 7:

The system of claim 1, where the tracking component includes a visual-tracking system.



## Claim 8:

The system of claim 7, where the visualtracking system includes one or more video cameras positioned within the facility.



## Claim 5:

The system of claim 1, where the authentication component includes a terminal configured to authenticate the user when a code provided to the terminal by the user matches a code stored on a token carried by the user.



#### Claim 9:

The system of claim 1, where the tracking component is configured to assess the users location within a grid imposed on the facility.



1006

1005

1009

1010

Claim 10:

The system of claim 9, where the control component is configured to compare the users location within the grid to one or more fixed grid locations associated with one or more of the touchpoints.



Claim 11:

The system of claim 1, where the control component is configured to include information identifying the user in the message delivered to the touchpoint.



Claim 12:

The system of claim 1, where the control component is configured to include an image depicting the user in the message delivered to the touchpoint.



1013