Abordagem Hierárquica com BERT para Classificação de Documentos Jurídicos Longos: Uma Análise de Técnicas de Agregação de Características

Jasson Carvalho da Silva[1] **, Ricardo Andrade Lira Rabelo** [1]**,**
Weslley Emmanuel Martins Lima[1] **,**
Vitor Augusto Correa Cortez Almeida[1]

[1]Departamento de Ciência da Computação - Universidade Federal do Piaui (UFPI)
Caixa Postal 64.049-550 – Teresina – PI – Brasi

jassonjcs11@gmail.com,{ricardoalr,weslley,vitor.cortez}@ufpi.edu.br

Abstract. *The study aims to evaluate and implement feature aggregation techniques in hierarchical models to deal with long texts in the legal domain. Transform-based models, although effective in many Natural Language Processing problems, face difficulties due to their quadratic complexity. due to quadratic complexity, especially with texts that exceed 512 tokens. To overcome these challenges, the study proposes investigating methods such as hierarchical models, text division strategies, local and sparse attention, with the aim of improving the processing and analysis of these texts, providing valuable insights for understanding and effective use in legal contexts.*

Resumo. *O estudo visa avaliar e implementar técnicas de agregação de características em modelos hierárquicos para lidar com textos longos no domínio jurídico. Modelos baseados em transformers, embora eficazes em vários problemas de Processamento de Linguagem Natural, enfrentam dificuldades devido à sua complexidade quadrática, especialmente com textos que excedem 512 tokens. Para superar esses desafios, o estudo propõe investigar métodos como modelos hierárquicos, estratégias de divisão de texto, atenção local e esparsa, com o objetivo de aprimorar o processamento e a análise desses textos, fornecendo insights valiosos para a compreensão e uso eficaz em contextos jurídicos.*

1. Introdução

Os textos jurídicos desempenham um papel crucial no sistema judiciário, assegurando uma justiça transparente e equitativa. Contudo, o processamento desses textos enfrenta desafios significativos devido à sua complexidade e extensão. Isso frequentemente resulta em atrasos na resolução de casos e na prestação de serviços judiciais, o que compromete a eficiência do sistema como um todo. Um exemplo preocupante dessa morosidade é a taxa de congestionamento processual líquido em primeiro grau, que atingiu 66,4% em 2023, conforme dados do Conselho Nacional de Justiça (CNJ)[Conselho Nacional de Justiça 2023]. Essa estatística revela que a maior parte dos processos permanece inconclusa no mesmo ano em que se inicia, sublinhando desafios críticos na gestão do volume de casos e na distribuição de recursos judiciais.

Este estudo visa explorar técnicas de inteligência artificial aplicadas à análise de textos jurídicos extensos. Em particular, este estudo investiga métodos para segmentar

textos complexos em unidades menores e integrar suas análises individuais para formar uma visão compreensiva do documento completo. Ao buscar melhorar a eficiência dessas técnicas, nosso objetivo é contribuir para avanços na precisão e na velocidade de processamento, proporcionando ferramentas mais eficazes para a análise jurídica.

Além disso, esta pesquisa não apenas aborda questões técnicas, mas também aspira a oferecer soluções práticas para os desafios enfrentados pelo sistema judiciário. Ao incorporar avanços em Processamento de Linguagem Natural (PLN), é esperado facilitar significativamente a análise de documentos jurídicos complexos. Ao eliminar a necessidade de abordagens manuais intensivas, como a leitura individual de cada documento, podemos ajudar juízes e advogados a trabalhar de maneira mais eficiente e precisa. A utilização estratégica dessas tecnologias pode não apenas acelerar os processos judiciais, mas também promover uma administração mais transparente e eficiente da justiça.

Ao longo deste trabalho, investigaremos como diferentes técnicas PLN voltadas especificamente para problemas de classificação textual podem ser aplicadas de maneira inovadora para aprimorar a compreensão e a utilização dos documentos jurídicos no contexto judicial contemporâneo.

2. Trabalhos relacionados

Dai et al. [Dai et al. 2022] exploraram técnicas hierárquicas com RoBERTa e Longformer, ajustando tamanhos de segmento e taxas de sobreposição para otimizar a eficácia em diversos conjuntos de dados. Entre as quatro bases investigadas, apenas uma era jurídica. Em Wagh et al. [Wagh et al. 2021] avaliaram modelos clássicos e baseados em *transformers* como BERT e DistilBERT, destacando limitações de conjunto de dados em relação ao contexto de documentos longos, não foram utilizadas bases do domínio jurídico. Chen et al. [Chen et al. 2022] introduziram o LordBERT, um modelo multitarefa que superou várias linhas de base em diferentes bases de dados, embora não tenha sido testado em conjuntos multi-rótulos.

Tuteja et al. [Tuteja and Juclà 2023] compararam múltiplos modelos em diversas bases de dados, empregando técnicas de truncamento e seleção de parágrafos importantes. Este estudo revela diferenças no desempenho entre modelos como o Longformer e alternativas mais simples, como *TF-IDF (Term Frequency - Inverse Document Frequency)* combinado com *DNN (Deep Neural Network)*, que oferecem treinamento mais rápido com desempenho comparável. Embora os textos analisados fossem longos, a pesquisa não abordou especificamente o contexto jurídico.

Os estudos mencionados abordaram uma variedade de textos, muitas vezes incluindo dados jurídicos, mas sem foco exclusivo nesse domínio. Este trabalho se distingue ao concentrar-se exclusivamente na análise de textos jurídicos longos, aplicando técnicas específicas para segmentar e integrar informações, proporcionando uma visão coesa do documento como um todo.

3. Metodologia

O método adotado neste trabalho consiste em cinco etapas: 3.1 Obtenção dos dados; 3.2 Implementação do modelo; 3.3 Geração de combinações de experimentos; 3.4 Execução de experimentos; 3.5 Avaliação de resultados. A Figura 1 apresenta os passo que serão melhores descritos nas seções abaixo.

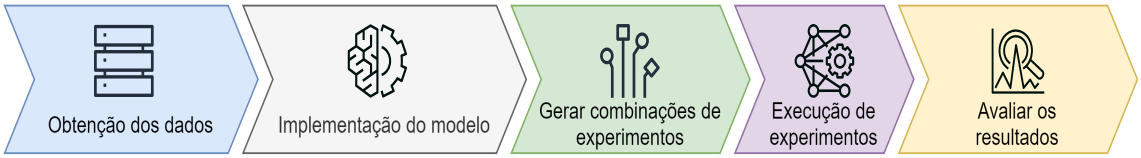

Figura 1. Fluxograma da metodologia adotada

3.1. Obtenção dos dados

Os experimentos foram realizados utilizando o conjunto de dados[1] LexGlue[Chalkidis et al. 2022], a base é formada por 7 conjuntos de dados compostos por ECtHR tarefa A e B, SCOTUS, EUR-LEX, LEDGAR, UNFAIR-ToS e CaseHOLD. Todos os conjuntos de dados são do domínio jurídico e são amplamente utilizados como bases de *benchmark*.

A ECtHR(A)[Chalkidis et al. 2019] contém 11.000 documentos da Corte Europeia de Direitos Humanos, divididos em 9.000/1.000/1.000 para treinamento, validação e teste respectivamente. Nesta tarefa, o modelo recebe uma lista de fatos de um caso e prevê os artigos da ECHR efetivamente violados em um cenário de multi-rótulo com 10 artigos selecionados de 66, permitindo que uma mesma entrada tenha de 0 a 10 rótulos de artigos violados. Em contraste, na tarefa B do mesmo conjunto, a saída esperada também é multi-rótulo, mas refere-se aos artigos supostamente violados, alterando o tipo de rótulo previsto.

O conjunto SCOTUS[Spaeth et al. 2020] é composto por 7.800 casos da Suprema Corte dos Estados Unidos, com 5.000/1.400/1.400 casos para treinamento, validação e teste respectivamente, dividido em 13 classes temáticas, como Direito Judicial e Federalismo. Este conjunto representa um problema de classificação multi-classe, onde cada caso é atribuído a exatamente uma das 13 classes possíveis.

A Figura 2 ilustra a distribuição de tokens para ECtHR(A) e SCOTUS, tokenizadas pelo BERT-based-uncased do Google[2]. A linha vertical cinza marca 512 tokens, acima do qual estão 82,40% dos documentos ECtHR(A) e 98,73% dos documentos SCOTUS. A linha azul marca 8192 tokens, com 3,12% dos documentos ECtHR(A) e 83,60% dos documentos SCOTUS acima desse limite. Para os experimentos, o limite máximo foi 8192 tokens.

3.2. Implementação do modelo

Os transformers hierárquicos adaptam a estrutura do BERT, focando a atenção em segmentos do documento individualmente. Agregadores são então usados para combinar as representações de cada segmento, formando uma codificação do documento completo.

[1] https://huggingface.co/datasets/coastalcph/lex_glue
[2] https://huggingface.co/google-bert/bert-base-uncased

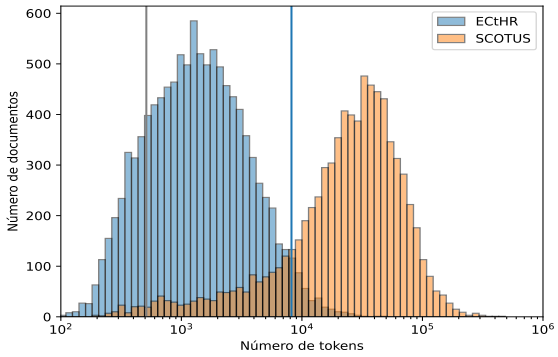

Figura 2. Distribuição de tokens por documento, utilizando o tokenizador do BERT-base-uncased.

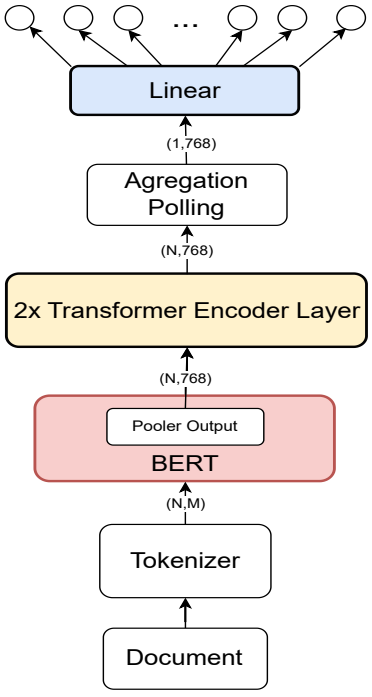

Figura 3. Estrutura da HAN utilizando a saída padrão do BERT para classificação, a *pooler output*.

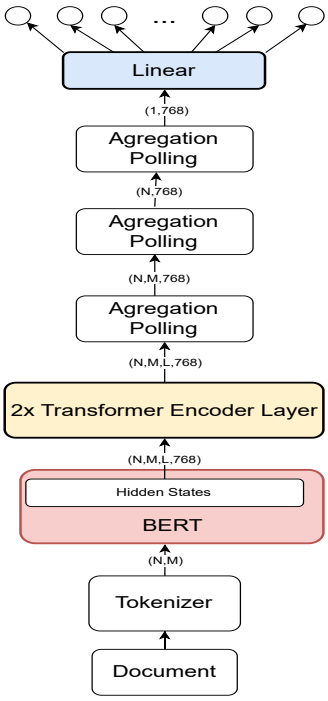

Figura 4. Estrutura da HAN utilizando a saída das camadas ocultas do BERT.

As figuras ilustram duas abordagens da rede: uma usa a saída padrão do BERT (*pooler output*) para classificação direta (Figura 3), e outra utiliza as camadas ocultas do BERT (*hidden states*) (Figura 4). A segmentação dos tokens é definida por parâmetros como tamanho máximo de segmento, número máximo de segmentos e sobreposição entre eles, garantindo que cada segmento capture o contexto circundante.

Essa estrutura é uma variante do HIERBERT [Chalkidis et al. 2021], que utiliza um codificador de segundo nível para contextualizar e agregar as representações geradas pelas camadas anteriores. A escolha da função de agregação (média, mediana ou máxima) depende da camada e do tipo de representação utilizados.

3.3. Geração de Combinações de Agregações

A Figura 4 ilustra a combinação de até três métodos de agregação para o mesmo modelo, podendo envolver o uso de *mean-pooling*, *max-pooling* ou *median-pooling*. Esses métodos foram avaliados inicialmente com a saída de *pooler predict*, reduzindo o número de experimentos viáveis devido a restrições de recursos. Posteriormente, a agregação de *max-pooling* foi excluída dos testes subsequentes por em ambas as bases apresentar um resultado inferior ao *baseline*.

Para o nível de agregação de camadas, foram explorados dois métodos principais: a Soma das Últimas Quatro Camadas[Devlin et al. 2018] e a Média a Nível de Camada. Essas escolhas resultaram em um total de 20 experimentos, permitindo uma análise abrangente das técnicas de *pooling* mais adequadas para a tarefa em questão.

3.4. Execução

Os experimentos foram realizados na plataforma vast.ai[3] com GPU de 16 GB de memória, limitando o tamanho de lote físico para 2. Os hiper-parâmetros foram escolhidos com base em estudos anteriores: [Chalkidis et al. 2022] influenciou o tamanho dos segmentos, quantidade total de tokens e funções de ativação, enquanto [Dai et al. 2022] orientou a taxa de sobreposição entre os segmentos.

No conjunto de dados ECtHR(A), foi empregado o otimizador AdamW com a função de perda BCEWithLogitLoss e função de ativação sigmoid. Para o conjunto de dados SCOTUS, utilizou-se o mesmo otimizador, mas com a função de perda CrossEntropyLoss e função de ativação softmax.

Foi analisado o impacto do tamanho do lote nos experimentos, testando tamanhos de lote de 8, 16 e 32. O tamanho físico do lote foi fixado em 2, com variação na estratégia de acumulação de gradientes. Isso significa que, independentemente do tamanho do lote considerado, a memória real utilizada por lote foi de 2, e as diferenças nos resultados se devem exclusivamente à forma como os gradientes foram acumulados. Os modelos com saída *pooler output* e *last hidden state* foram avaliados, aplicando agregação média ao nível de segmento para *pooler output*, e agregações de média tanto ao nível de segmento quanto de token para *last hidden state*.

3.5. Avaliação

Foram usadas as métricas micro-F1(μ-F1) e macro-F1(m-F1) para avaliar o modelo. A μ-F1 combina a precisão e o recall de todas as classes como se fossem uma única classe, sem considerar o desbalanceamento das classes. Já a m-F1 calcula a média do F1 de cada classe individualmente, sendo sensível a desbalanceamentos, pois dá igual peso a cada classe, destacando problemas com classes minoritárias.

[3] `https://vast.ai`

4. Resultados e Discussão

Utilizou-se o modelo BERT-base-cased na base de dados ECtHR(A), focando na tarefa A. Foi observado que o tamanho de lote de 32 consistentemente superou os tamanhos de lote de 8 e 16 em termos de μ-F1 e m-F1. Em média, foram observadas melhorias de 2,75% em relação ao lote de 16 e 25,74% em relação ao lote de 8 para todas as métricas avaliadas. Esses resultados enfatizam a importância crítica da escolha adequada do tamanho de lote para otimizar tanto a eficiência computacional quanto a qualidade das métricas de avaliação.

Os experimentos foram conduzidos conforme as especificações detalhadas para os conjuntos de dados ECtHR(A) e SCOTUS. A arquitetura HAN com 115 milhões de parâmetros foi implementada em ambos os casos. Cada época dos experimentos com HAN levou aproximadamente 30 minutos para ECtHR(A) e 52 minutos para SCOTUS, enquanto o baseline consumiu cerca de 16 minutos por época para ECtHR(A) e 9 minutos para SCOTUS. O uso de memória da GPU variou entre 12,3 GB e 14,7 GB, dependendo do modelo e conjunto de dados, suportando até 8192 tokens por documento, com um máximo de sequência de 128 tokens.

Os modelos foram treinados com um lote físico de 2 documentos, utilizando uma estratégia de acumulação de gradientes de 16, resultando em 32 documentos por lote. A sobreposição de segmentos foi fixada em 25%, conforme estudos de [Dai et al. 2022] e [Chalkidis et al. 2022]. O treinamento foi interrompido após 5 épocas, com base na perda de validação. Essas configurações foram escolhidas para maximizar a eficiência computacional e garantir a estabilidade dos modelos durante o treinamento, seguindo metodologias reconhecidas no processamento de linguagem natural aplicado a documentos jurídicos extensos.

4.1. Resultados experimentais

Esta seção apresenta os resultados dos experimentos realizados para avaliar as diferentes estratégias de agregação aplicadas ao modelo BERT em tarefas de classificação jurídica. Os resultados são discutidos com base em duas abordagens principais: a saída da camada de classificação do BERT (*pooler output*) e as camadas ocultas (*hidden states*). A análise inclui comparações entre métodos de agregação e suas performances nas bases de dados ECtHR(A) e SCOTUS.

4.1.1. Pooler Predict

A Tabela 1 exibe o comparativo entre os diferentes tipos de agregação testados utilizando a saída da camada de classificação do BERT, a *pooler output*. A *pooler output* é a camada comumente usada para tarefas de classificação. A representação obtida através dela é a representação do token [CLS] depois de processado por uma camada linear e uma função de ativação tanh.

Os métodos de agregação *mean-pooling*, *median-pooling* e *max-pooling* foram comparados em termos de desempenho em diferentes conjuntos de dados, na ECtHR(A), o *mean-pooling* alcançou a melhor m-F1, enquanto o *median-pooling* foi superior em μ-F1. Já na SCOTUS, o *mean-pooling* superou ambos em ambas as métricas. O *max-*

Tabela 1. Os resultados comparativos incluem *max-pooling*, *mean-pooling*, *median-pooling* e a baseline first-512 usando a estrutura HAN com codificador de segundo nível.

Base de Dados	***max-pooling***		***mean-pooling***		***median-pooling***		**First-512**	
	μ-F1	m-F1	μ-F1	m-F1	μ-F1	m-F1	μ-F1	m-F1
ECtHR(A)	58,81	43,34	69,08	**56,21**	**69,47**	53,50	63,91	47,60
SCOTUS	60,04	40,10	**68,04**	**53,49**	58,93	33,48	64,80	47,09

pooling teve um desempenho inferior na ECtHR(A) e, embora tenha superado o *median-pooling* na SCOTUS, não alcançou o baseline em nenhum dos casos. Esses resultados destacam a variabilidade dos métodos de pooling e sua sensibilidade ao contexto específico do conjunto de dados e das métricas avaliadas.

A escolha entre *median-pooling* e *mean-pooling* depende da métrica a ser destacada, especialmente em bases de dados desbalanceadas. A *median-pooling*, menos sensível a *outliers*, pode ter desempenho inferior em m-F1, pois trata todas as instâncias de forma equitativa, o que pode não capturar bem as nuances das classes minoritárias. Em contraste, a *mean-pooling*, mais sensível a *outliers*, tende a proporcionar uma representação mais equilibrada das classes, resultando em melhor desempenho em m-F1. Assim, *mean-pooling* é geralmente recomendado para agregação a nível de segmento em cenários com classes desbalanceadas.

4.1.2. Hidden States

O uso das camadas *hidden state* adiciona níveis adicionais de agregação à classificação, incorporando todos os tokens na representação final. Uma única camada entre as 12 camadas ocultas aplica uma agregação a nível de token, enquanto múltiplas camadas aplicam uma agregação adicional entre camadas. Em contraste, a *pooler output* utiliza apenas a representação do token [CLS].

A Tabela 2 compara diferentes abordagens de uso das *hidden states*, incluindo agregações de média (mean) e mediana (median) a nível de segmento. A abordagem *last-four-state* adiciona a agregação de soma (sum) das últimas quatro camadas ocultas, consideradas as mais semânticas. Estudos como o de [Jawahar et al. 2019] destacam que o BERT encapsula múltiplos níveis de informações linguísticas, camadas mais baixas capturam informações superficiais e camadas superiores agregam informações sintáticas e semânticas.

4.2. Escolha entre Pooler Predict ou Last Hidden State

O uso de média (mean) nas camadas ocultas demonstrou ser superior em todos os casos em relação ao uso da *pooler output*.No conjunto ECtHR(A), *mean-mean* obteve um resultado até 1,30% melhor comparado ao uso de *mean-pooling* com *pooler output*, o mesmo ocorre com a *mean-median* tendo um desempenho 0,81% superior a sua contraparte, o mesmo comportamento ocorre na base de dados SCOPUS, a tabela 3 apresenta cada um desses resultados.

Tabela 2. Resultados para o modelo BERT na tarefa ECtHR(A) com diferentes estratégias de agregação em hidden state. Lê-se layer-token-segmento acerca dos níveis de agregação mostrados.

Base de Dados	**last-hidden-state**				**last-four-state**			
	mean-mean		**mean-median**		**sum-mean-mean**		**mean-mean-mean**	
	μ-F1	m-F1	μ-F1	m-F1	μ-F1	m-F1	μ-F1	m-F1
ECtHR(A)	69,34	**57,59**	69,40	54,57	70,34	55,11	**71,44**	56,51
SCOTUS	67,83	54,69	64,97	49,85	**69,06**	55,98	67,42	**56,04**

Tabela 3. Comparação de métricas entre PO (pooler output) e LHS (last hidden state). O valor representado é a média de μ-F1 e m-F1 para cada caso.

	Média		**Mediana**	
Base de Dados	**LHS**	**PO**	**LHS**	**PO**
ECtHR(A)	**63,46**	62,64	**61,98**	61,48
SCOTUS	**61,26**	60,76	**57,41**	46,20

O uso das camadas ocultas provou ser mais eficaz que a saída da *pooler output* para tarefas de classificação. Embora o token [CLS] capture o contexto do segmento, sua eficácia é limitada. A agregação de média, que considera todos os tokens, mostrou melhor desempenho. Comparando os modelos das últimas quatro camadas com aqueles que usam apenas a última, como mostrado na Tabela 2, destacam essa superioridade. Considerando a média entre μ-F1 e m-F1, em ambos os conjuntos de dados, ECtHR(A) e SCOTUS, os modelos que exploraram as últimas quatro camadas obtiveram os melhores resultados: '*sum-mean-mean*' para SCOTUS e '*mean-mean-mean*' para ECtHR(A). Esses resultados evidenciam o potencial das camadas mais profundas na captura de uma representação mais rica e detalhada dos dados, resultando em um desempenho superior nas tarefas de classificação.

4.3. Tempo de Convergência

Os modelos que utilizam *hidden states* demonstraram uma convergência mais rápida nos conjuntos de dados ECtHR(A) e SCOTUS, terminando o treinamento em menos épocas comparados aos modelos que usam *pooler output*. Além da eficiência na convergência, os modelos de *hidden states* alcançaram melhores resultados de μ-F1 e menor perda, indicando uma aprendizagem mais eficaz dos padrões nos dados. Em contraste, as abordagens baseadas em *pooler output* mostraram uma convergência ligeiramente mais lenta e, em alguns casos, um desempenho inferior.

Esse cenário destaca a importância crítica da escolha de uma técnica de agregação para otimizar a eficiência do modelo. Modelos que utilizam *hidden states* não apenas convergem mais rapidamente, mas também mantêm um desempenho competitivo em comparação com outras abordagens, como evidenciado pela análise comparativa dos resultados. Reduzindo os custos computacionais associados ao treinamento.

4.4. Influência dos níveis de agregação

A Tabela 4 compara os efeitos da agregação média e mediana em níveis de segmento e token nas bases de dados ECtHR(A) e SCOTUS. Na ECtHR(A), a agregação median-mean destacou-se com μ-F1 de 69,88%, enquanto mean-median teve um desempenho inferior em m-F1 (54,57). Na SCOTUS, median-mean também foi superior, alcançando os melhores resultados em μ-F1 (69,38) e m-F1 (61,66), em comparação com mean-median, que obteve μ-F1 de 64,97%.

Esses resultados indicam que a agregação median-mean é mais robusta e eficaz para maximizar o desempenho em μ-F1 e m-F1, enquanto mean-median pode ser menos eficaz em certos contextos. A escolha da função de agregação a nível de token tem um impacto maior no desempenho do modelo do que a agregação a nível de segmento. Mean-mean mostrou resultados consistentes, enquanto mediana a nível de segmento resultou em métricas inferiores. No entanto, mediana a nível de token alcançou resultados semelhantes aos de mean-mean.

Portanto, a escolha da função de agregação deve considerar as características específicas da base de dados e as métricas de desempenho prioritárias. Median-mean se mostrou mais robusta para SCOTUS em todas as métricas e para ECtHR(A) em μ-F1, destacando-se como a melhor opção globalmente. Integrar agregação a nível de camada pode potencializar ainda mais os resultados de median-mean. É crucial considerar esses aspectos na seleção da função de agregação ideal, dada a ausência de um método superior único que se aplique uniformemente a ambas as bases de dados.

Tabela 4. Comparativo entre média e mediana usadas em níveis de agregação diferentes.

Base de Dados	**last-hidden-state**					
	mean-mean		**mean-median**		**median-mean**	
	μ-F1	m-F1	μ-F1	m-F1	μ-F1	m-F1
ECtHR(A)	69,34	**57,59**	69,40	54,57	**69,88**	54,37
SCOTUS	67,83	54,69	64,97	55,98	**69,38**	**61,66**

5. Conclusão

Este estudo explorou técnicas avançadas de Processamento de Linguagem Natural (PLN) para análise eficiente de documentos jurídicos extensos, utilizando modelos hierárquicos e estratégias de agregação de características. Modelos baseados em *hidden states* demonstraram melhor desempenho e convergência mais rápida em comparação com abordagens que utilizam *pooler output*, economizando recursos computacionais durante o treinamento. Técnicas de agregação como média e mediana mostraram-se consistentes, impactando positivamente os resultados finais.

O estudo abordou três principais questões: a eficácia das técnicas de agregação, a eficiência na análise de documentos jurídicos e os requisitos computacionais necessários. Constatou-se que variações com média e mediana junto com *hidden states* são mais eficazes para textos jurídicos longos. Não foi encontrada uma única combinação com resultado superior em todas as bases, no entanto variações com mediana e média costumam

ter os melhores resultados, o uso de quantidades maiores de níveis de agregação obtêm um resultado melhor que utilizar apenas um(*pooler predict*) ou dois nivels(*last hidden state*)

Por outro lado, o modelo Hierarchical Attention Network (HAN) enfrentou desafios significativos devido à sua complexidade e exigência de recursos computacionais. Limitações de memória e orçamentárias restringiram o tamanho do lote físico e a exploração de modelos mais complexos. Para trabalhos futuros, propõe-se superar essas limitações e fazer uso de modelo de atenção esparsa como Longformer e BigBird.

Referências

[Chalkidis et al. 2019] Chalkidis, I., Androutsopoulos, I., and Aletras, N. (2019). Neural legal judgment prediction in English. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4317–4323, Florence, Italy. Association for Computational Linguistics.

[Chalkidis et al. 2021] Chalkidis, I., Fergadiotis, M., Tsarapatsanis, D., Aletras, N., Androutsopoulos, I., and Malakasiotis, P. (2021). Paragraph-level rationale extraction through regularization: A case study on european court of human rights cases. *arXiv preprint arXiv:2103.13084*.

[Chalkidis et al. 2022] Chalkidis, I., Jana, A., Hartung, D., Bommarito, M., Androutsopoulos, I., Katz, D., and Aletras, N. (2022). LexGLUE: A benchmark dataset for legal language understanding in English. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4310–4330, Dublin, Ireland. Association for Computational Linguistics.

[Chen et al. 2022] Chen, B., Sun, R., Dai, Y., Zheng, H.-T., and Zhang, R. (2022). Lordbert: Embedding long text by segment ordering with bert. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

[Conselho Nacional de Justiça 2023] Conselho Nacional de Justiça (2023). Justiça em números 2023.

[Dai et al. 2022] Dai, X., Chalkidis, I., Darkner, S., and Elliott, D. (2022). Revisiting transformer-based models for long document classification. *arXiv preprint arXiv:2204.06683*.

[Devlin et al. 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Jawahar et al. 2019] Jawahar, G., Sagot, B., and Seddah, D. (2019). What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

[Spaeth et al. 2020] Spaeth, H. J., Epstein, L., Segal, J. A., Martin, A. D., Ruger, T. J., and Benesh, S. C. (2020). Supreme court database, version 2020 release 01. `http://supremecourtdatabase.org`. Accessed: 2024-06-02.

[Tuteja and Juclà 2023] Tuteja, M. and Juclà, D. G. (2023). Long text classification using transformers with paragraph selection strategies. In *Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 17–24.

[Wagh et al. 2021] Wagh, V., Khandve, S., Joshi, I., Wani, A., Kale, G., and Joshi, R. (2021). Comparative study of long document classification. In *TENCON 2021-2021 IEEE Region 10 Conference (TENCON)*, pages 732–737. IEEE.