Proposta recebida em September 2025 e aceite para publicação em November 2025.

Aprendizado por Transferência para Correção Automática de Redação

Transfer Learning for Automatic Essay Scoring

Igor Cataneo Silveira
INESC-ID Lisboa, Portugal
Universidade de São Paulo, Brasil

Eugénio Ribeiro
INESC-ID Lisboa, Portugal
Instituto Universitário de Lisboa (ISCTE-IUL), Portugal

Nuno Mamede
INESC-ID Lisboa, Portugal
Instituto Superior Técnico, Universidade de Lisboa, Portugal

Jorge Baptista
INESC-ID Lisboa, Portugal
Faculdade de Ciências Humanas e Sociais, Universidade do Algarve, Portugal

Resumo

A tarefa de Correção Automática de Redação tem despertado crescente interesse na área de processamento de texto em português. Entre os conjuntos de dados disponíveis, destaca-se um corpus de redações narrativas produzidas por alunos do 5º ao 9º ano do ensino fundamental no Brasil. Essas redações são avaliadas segundo quatro competências: registro formal, coerência temática, estrutura retórica narrativa e coesão textual. Este trabalho explora a criação de um sistema baseado em conhecimentos derivados de outro *dataset* (desenvolvido com base em textos produzidos para o ENEM) e de outras tarefas (cálculo de complexidade textual e análise de legibilidade). O sistema desenvolvido combina modelos neurais, características (*features*) curadas calculadas por programas de análise textual e seleção de *features* em um modelo de Aprendizado em Dois Estágios. Com isso, foi possível elevar a *performance* em relação ao estado-da-arte, nomeadamente, em 9% para a primeira competência, 5,5% para a terceira e 8,9% para a quarta.

Palavras chave

correção automática de redações, textos narrativos, língua portuguesa

Abstract

Automatic Essay Scoring is a field that has been receiving a lot of attention in Portuguese. Among the available datasets, one stands out: a corpus of narrative essays written by students from 5th to 9th grade in Brazil. These essays were evaluated according to four traits: formal register, thematic coherence, narrative rhetorical structure, and textual cohesion. This work explores the development of a system based on knowledge from another dataset (developed from texts produced for the Brazilian national entrance exam, ENEM) and from other tasks (textual complexity and legibility analysis). This developed system combines neural models, handcrafted features calculated by textual analysis software, and feature selection, through a Two Stage Learning algorithm. With this system, the state-of-the-art performance was enhanced by 9% for the first trait, 5.5% for the third, and 8.9% for the fourth one.

Keywords

automatic essay scoring, narrative essays, Portuguese language

1. Introdução

Correção Automática de Redação é uma tarefa da Inteligência Artificial e do Processamento de Linguagem Natural que tem como objetivo tornar as máquinas capazes de corrigir produções textuais segundo critérios definidos previamente. O progresso dessa tarefa pode se dar em três dimensões principais: (1) quais aspectos textuais que estão sendo corrigidos; (2) quão bem os aspectos propostos estão sendo corrigidos; (3) o quanto o sistema pode ajudar a melhorar o texto submetido. Se um sistema fosse capaz de avaliar — com a precisão de um professor qualificado — a ortografia, a gramática, a coesão, a coerência e a adequação ao tema, e ainda fosse capaz de sugerir melhorias, certamente esse sistema impactaria significativamente o processo educacional.

A implementação de um sistema tão completo, porém, ainda não foi realizada. Isso acontece por diversas razões. Primeiramente, porque diferentes aspectos textuais exigem, potencialmente, diferentes técnicas de Processamento de Linguagem Natural. Por exemplo, ortografia pode ser

DOI: 10.21814/lm.17.2.493

corrigida com uma lista — eventualmente bastante extensa — de palavras. Contudo, essa mesma abordagem não é capaz de corrigir coesão. Em segundo lugar, cada dimensão mencionada no parágrafo anterior é um problema distinto que exige conhecimentos diferentes. Consideremse, por exemplo, as dimensões da qualidade da correção e da capacidade de o sistema de contribuir para a melhoria do texto: na primeira são discutidas métricas de comparação entre dois anotadores (Yannakoudakis & Cummins, 2015), enquanto na outra pode-se discutir a efetividade (Wilson & Czik, 2016) do *feedback*.

Outra dificuldade da área é a falta de grandes *datasets*. Em inglês, o *dataset* mais famoso, ASAP (Hamner et al., 2012), possui 8 subconjuntos distintos contendo, cada um, aproximadamente 1.500 redações. Cada um desses subconjuntos possui um tema e formato diferentes, além de diretrizes diferentes para as avaliações. Em português brasileiro, o maior *dataset* deriva de redações do Exame Nacional do Ensino Médio (ENEM) (Marinho et al., 2022a), e contém 6.579 redações — os critérios de avaliação são os mesmos, mas cada tema reúne um número relativamente limitado de redações.

Recentemente foi lançada uma competição para correção de redações narrativas (Mello et al., 2024). Este artigo utiliza o conjunto de dados disponibilizado nessa competição. Neste *dataset*, 1.235 redações escritas por alunos do ensino fundamental foram avaliadas por humanos segundo quatro aspectos, chamados de *competências*: Registro Formal, Coerência Temática, Estrutura da Retórica Narrativa e Coesão Textual. Cada competência é avaliada com uma nota de um a cinco; o conjunto de dados foca exclusivamente nessas pontuações, sem incluir anotações de *feedback*.

A abordagem aqui seguida para corrigir as redações é baseada em transferência de conhecimento de outras tarefas para a correção de redações narrativas. Para isso, desenvolveu-se um algoritmo de Aprendizado em Dois Estágios (Liu et al., 2019) com seleção de *features*. Essa abordagem pode ser dividida em quatro partes: Inicialmente, (1) se selecionou um modelo BERTimbau (Souza et al., 2020) pré-treinado para correção em outro *dataset* (Silveira et al., 2024) e se fez *fine-tuning* no *dataset* atual. De seguida, (2) se testaram três diferentes conjuntos de *features*, extraídos a partir de:

(a) o NILC-Metrix (Leal et al., 2024), que não foi desenvolvido para a tarefa correção de redação;

(b) o iRead4Skills[1], que é utilizado para análise de legibilidade;

(c) modelos neurais treinados na etapa anterior;

Depois, (3) agrupamos todas as *features* anteriores; Por fim, (4) utilizamos técnicas de seleção de *features* para encontrar um subgrupo que maximize a *performance* do sistema. Com isso, foi possível elevar a *performance* acima do estado-da-arte em 9% para a primeira competência, 5,5% para a terceira e 8,9% para a quarta. Apesar de a *performance* não ter aumentado na segunda competência, ela ainda ficou competitiva contra os modelos da competição.

As contribuições deste trabalho são:

1. Mostra-se que é possível utilizar o *dataset* AES-ENEM para pré-treinar modelos e corrigir redações do *dataset* de texto narrativo;
2. Mostra-se que, no *dataset* narrativo, algoritmos mais simples baseados em *features* são competitivos perante modelos mais complexos;
3. Mostra-se a efetividade de se fazer seleção de *features* em um sistema de Aprendizado em Dois Estágios;
4. Aumentou-se a *performance* estado-da-arte em três das quatro competências.
5. Apresentam-se as *performances* também separadas por tema.
6. Disponibilizam-se os modelos neurais no HuggingFace[2] e os outros *scripts* no github[3].

O restante do artigo é dividido da seguinte forma: Os trabalhos relacionados são comentados na Seção 2. Em seguida, discutiremos nossa metodologia na Seção 3. Os resultados, etapa a etapa, serão apresentados na Seção 4. Por fim, os resultados são discutidos na Seção 5, e as conclusões apresentadas na Seção 6.

2. Trabalhos Relacionados

A tarefa de Correção Automática de Redação em língua portuguesa tem atraído bastante atenção nos últimos anos. Esta seção discute os algoritmos usados recentemente, com foco especial nos que participaram da competição de redações narrativas (Mello et al., 2024).

[1] https://gitlab.hlt.inesc-id.pt/iread4skills/docs

[2] https://huggingface.co/igorcs/models.

[3] https://github.com/SilveiraIgor/NarrativeEssays-Experiments.

Até o momento, são conhecidos quatro conjuntos de dados disponíveis para Correção Automática de Redação em português. O primeiro foi apresentado em Amorim & Veloso (2017), contendo redações de treinamento para o ENEM extraídas da internet. Para rotular as redações presentes nesse *dataset*, utilizou-se regressão linear sobre 19 *features* adaptadas do inglês e de um domínio específico. Os resultados, medidos pelo Kappa Quadrático (QWK), variaram entre 0,15 e 0,33 nas competências e alcançaram 0,42 na nota final. Nesse estudo, também se empregam características, algumas calculadas por ferramentas que extraem métricas textuais gerais. Adicionalmente, se faz ainda seleção de *features* para melhorar o desempenho (*performance*).

Em seguida, foram publicados o *dataset* Essay-br e sua versão estendida (Marinho et al., 2021, 2022a). O corpus também é composto por textos extraídos da internet, provenientes de dois sites distintos e com notas normalizadas de modo a corresponder ao padrão oficial de avaliação. Para este *dataset* já foram testadas diversas abordagens, desde algoritmos de *features* escolhidas especialmente para essa tarefa até LSTMs (Marinho et al., 2022b). Os melhores resultados para cada competência variaram entre 0,46 e 0,56 — medido em QWK. Nas duas primeiras competências, o algoritmo baseado em *features* apresentou melhor desempenho, enquanto que nas três seguintes foram os baseados em LSTM. Os autores argumentam que as duas primeiras competências são menos subjetivas, sendo então mais fácil definir *features* relevantes. O presente trabalho difere desse estudo por usar as mesmas *features* para todas as competências, as quais não foram escolhidas especificamente para o nosso *dataset* e, ainda por serem combinadas com *features* extraídas de Grandes Modelos de Linguagem.

Posteriormente, o conjunto de dados AES-ENEM (Silveira et al., 2024) baseou-se nas mesmas fontes, reorganizando as redações conforme a origem e disponibilizando os arquivos HTML das respectivas páginas, além de adicionar outras informações, tais como comentários do corretor. Disponibilizou, ainda, correções adicionais para uma das fontes, as quais foram feitas por dois especialistas independentes. Foi mostrado que a concordância entre os especialistas era maior entre eles do que com as notas disponibilizadas na internet. Os autores testaram diferentes configurações do modelo BERTimbau (Souza et al., 2020), medindo as *performances* de acordo com quatro métricas: acurácia, raiz do erro quadrático, QWK e divergência (métrica utilizada pelo próprio ENEM para dizer que dois avaliadores deram notas incompatíveis). Todos os modelos treinados foram disponibilizados e alguns são utilizados neste trabalho. Silveira et al. (2025) sustentam que, para redações que simulam o ENEM, a regressão linear com *features* extraídas pelo NILC-Metrix (Leal et al., 2024) pode alcançar desempenho semelhante ao de algoritmos baseados em Transformers. Este trabalho explora o uso de *features* extraídas tanto pelo NILC-Metrix, quanto pelo iRead4Skills e pelo BERTimbau. Além disso, são testados outros preditores além do regressor linear.

Para a competição de avaliação de redações narrativas (Mello et al., 2024), foram testadas sete abordagens, das quais três delas são métodos *baseline*. O primeiro dos *baselines* é baseado em calcular o TF-IDF das redações e usar essas representações como entrada para algoritmos de aprendizado tradicionais, disponíveis no scikit-learn (Pedregosa et al., 2011) com adição do XGBoost (Chen & Guestrin, 2016) e LGBM (Ke et al., 2017). O segundo *baseline* usa as representações geradas por um BERTimbau — sem *fine-tuning* para essa tarefa — como entrada para os mesmos algoritmos de aprendizado utilizados no *baseline* anterior. O terceiro *baseline* é um *fine-tuning* do BERTimbau como classificador.

As quatro outras abordagens foram de participantes na competição e são apresentadas como AESVoting (de Lima et al., 2024), INESC-ID (Ribeiro et al., 2024), PiLN (de Sousa et al., 2024) e Ocean Team. O primeiro participante representou as redações por meio de vetores densos obtidos com o BERTimbau — semelhante ao segundo *baseline* — e, então, utilizou três classificadores, sendo a nota final decidida através do voto da maioria. O segundo participante fez o *fine-tuning* do Albertina (Rodrigues et al., 2023) e do BERTimbau-large. O terceiro participante se baseou no BERTimbau como regressor em vez de classificador. Adicionalmente, também fez com que o mesmo modelo gerasse as notas das quatro competências ao mesmo tempo. Por fim, o último modelo do quarto participante também gera representações usando o BERTimbau, mas o TF-IDF da redação foi ainda concatenado à representação anterior e o vetor resultante é passado para um algoritmo de aprendizado tradicional.

Todos os trabalhos elencados anteriormente podem ser vistos como representando uma etapa do modelo de Aprendizado em Dois Estágios (Liu et al., 2019). Esse modelo, originalmente aplicado a um conjunto de redações em inglês, gera inicialmente notas com base em modelos neurais. No estágio seguinte, essas notas são adicionadas a

um conjunto de outras seis *features* selecionadas manualmente. O conjunto final é então usado por um XGBoost que prediz a nota final. A abordagem proposta neste estudo inspira-se nesse modelo, ampliando o uso de características derivadas de modelos neurais e com um maior conjunto de *features* curadas. Por isso, há também um estágio de seleção de *features* para melhorar a *performance*.

3. Metodologia

Esta seção descreve o conjunto de dados de redações narrativas apresentado por Mello et al. (2024), doravante designado apenas por DEF (*Dataset* de Ensino Fundamental), e as métricas usadas na competição em que ele foi apresentado. Em seguida, serão comentados os modelos e as *features* curadas aqui utilizadas. Por fim, descrevem-se as quatro etapas do processo.

3.1. *Dataset*

O DEF é composto por 1.235 redações narrativas escritas por alunos do 5º ao 9º ano de escolaridade, em escolas públicas do Brasil. Cabe observar que os dados relativos ao ano escolar do aluno que escreveu a redação não estão disponíveis. O corpus dessas redações foi dividido em treinamento, validação e teste, contendo, respectivamente, 740, 125 e 370 exemplos.

As redações foram escritas para duas propostas diferentes. A primeira é sobre ter pintado animais na parede do quarto e essas pinturas terem desaparecido depois de um tempo, enquanto a segunda é sobre encontrar uma pedra muito brilhante. Todas as divisões do *dataset* possuem exemplos das duas propostas, mas em proporções diferentes — 3% das redações do treinamento são do tema da pedra, na validação são 6% e no teste são 4%. As redações foram avaliadas por profissionais, segundo quatro competências distintas:

C1: Registro formal — ou bom uso do português;

C2: Coerência temática — ou quão plausível é o texto de acordo com a proposta;

C3: Estrutura Retórica Narrativa — ou a adequação do texto à tipologia narrativa;

C4: Coesão Textual — ou bom uso dos mecanismos linguísticos para conectar diferentes partes do texto.

Cada competência foi avaliada com uma nota, correspondente a um número inteiro, entre 1 e 5.

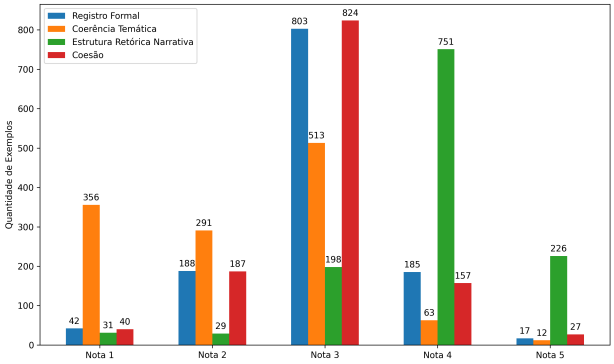


Figura 1: Número de redações com cada uma das notas em cada competência (C). C1 em azul, C2 em laranja, C3 em verde e C4 em vermelho. Adaptado de Mello et al. (2024).

A distribuição de notas por competência pode ser vista na Figura 1. Observa-se que as competências apresentam distribuições distintas: C1 e C4 seguem padrões aproximadamente normais, com média em 3 e com baixo desvio-padrão. A competência C3 tem um padrão semelhante, mas com média em 4, enquanto a C2 não se parece com nenhuma distribuição. Distribuições tão desiguais podem introduzir vieses significativos nos modelos, bem como nas métricas de *performance*.

3.2. Métricas

Na competição, os modelos foram avaliados de acordo com duas métricas, a F1 Ponderada e o Coeficiente de Kappa — o qual chamaremos apenas de Kappa (κ). A F1 Ponderada varia de 0 a 1, os extremos indicam a pior e a melhor *performance* possível, respectivamente. Para as competências C1, C2 e C4, é possível conseguir valores razoáveis nesta métrica apenas dizendo a classe mais frequente, já que assim terá *recall* perfeito (para essa classe) e uma precisão alta, já que a classe em questão também representa a maior parte do *dataset*.

Por outro lado, um modelo que atribua sempre a mesma classe obtém valor zero na métrica κ. Isso se dá pelo fato de essa métrica variar de -1 até 1, onde -1 significa discordância total, 1 significa concordância total e zero significa concordância aleatória. Convém observar que as duas métricas estão, de certo modo, associadas, uma vez que ambas estão relacionadas com a acurácia tradicional. Além disso, é impossível maximizar uma delas sem maximizar a outra. Portanto, é de se esperar que, conforme a acurácia do modelo suba, as duas métricas também melhorem — alternativamente, conforme uma das métricas vá se aproximando de 1, a outra também se aproximará.

Portanto, neste trabalho optou-se por apresentar a soma das duas métricas ao longo da descrição dos modelos, reservando apenas para o final a exposição das métricas individualmente, de modo a possibilitar a comparação com trabalhos anteriores. Dessa forma, a *performance* será avaliada — na maior parte do estudo — por meio de uma métrica que varia entre -1 e 2, dado que os piores resultados possíveis são -1 e 0, enquanto os melhores correspondem a 1 e 1, respectivamente. Outra razão para esta opção prende-se com o fato de que, havendo duas métricas, seria potencialmente necessário efetuar comparações constantes em função de cada uma delas. A soma das duas métricas simplifica a comparação nas etapas intermediárias, pois seu aumento implica melhora em pelo menos uma delas.

3.3. Modelos

Neste trabalho, se utilizam alguns modelos neurais baseados no BERTimbau-base, alguns dos quais foram treinados unicamente com as redações do *dataset* narrativo, enquanto outros já haviam sido previamente treinados para correção de redação. Esses modelos, porém, foram treinados para corrigir redações do ENEM e foram disponibilizados por Silveira et al. (2024). Optamos por utilizar esses modelos porque eles já estavam disponíveis e porque, embora tenham sido (pré-)treinados com menos dados, esses modelos utilizaram conjuntos de dados de maior qualidade, provenientes de especialistas. Para treiná-los, foram utilizadas diferentes funções de perda — baseadas em regressão, classificação ou regressão ordinal — para cada uma das competências. Esses modelos são aqui utilizados pois se nota que existe uma proximidade muito grande entre as quatro primeiras competências do ENEM e as quatro competências do DEF.

Mais especificamente, das cinco competências do ENEM, a primeira é sobre o bom uso do português — equivalente à competência utilizada aqui. A segunda é sobre a adequação da redação ao tema proposto — semelhante ao que é utilizado aqui, mas no ENEM é um tema argumentativo, aqui é um tema narrativo. A terceira trata da estrutura retórica dissertativa, conceito aqui transposto para o gênero narrativo. A quarta competência, tanto no ENEM quanto aqui, é sobre a coesão existente no texto. Por fim, a quinta competência do ENEM avalia a proposta de solução, aspecto inexistente nas redações narrativas. Pelas razões previamente elencadas, se acredita fazer sentido realizar *fine-tuning* de modelos treinados para o ENEM no *dataset* deste estudo.

Além dos modelos neurais previamente mencionados, empregaram-se modelos tradicionais — não neurais — de aprendizado de máquina, tais como Regressão Linear, Florestas Aleatórias e XGBoost. Todos esses modelos, exceto o último, foram utilizados através da biblioteca scikit-learn (Pedregosa et al., 2011). Os três foram escolhidos por representarem graus diferentes de sofisticação: Regressão Linear é o preditor mais simples e estável, enquanto o XGBoost é mais complexo e dependente dos parâmetros; Florestas Aleatórias se situam no meio termo entre os dois.

Esses modelos receberam como entrada *features* selecionadas de três fontes distintas. Primeiro, dos próprios modelos neurais, são elas: a nota que o modelo daria; e/ou, para o caso do classificador, o softmax dos logits para cada classe. As outras duas fontes de *features* são provenientes de programas de análise textual.

O primeiro conjunto de *features* provém do NILC-Metrix (Leal et al., 2024), ferramenta voltada à avaliação de coesão, coerência e complexidade textual. Essa ferramenta calcula 72 características, incluindo proporções de classes gramaticais, ambiguidade lexical, extensão textual e índice de Flesch. O segundo conjunto de características provém do iRead4Skills, ferramenta originalmente desenvolvida para avaliar a legibilidade de textos em português europeu. Este conjunto de *features* é dividido em quatro categorias: descritivas, lexicais, sintáticas e discursivas. Ao todo, a ferramenta calcula 656 *features*.

3.4. Etapas

O estudo se encontra organizado em cinco etapas. Na primeira, se busca responder à seguinte questão: Será que os modelos previamente treinados podem ser usados como ponto de partida para outros *datasets* com critérios semelhantes? Foi utilizada uma configuração-padrão para realizar esse experimento: otimizador SGD tradicional com taxa de aprendizado 10^{-5} e 0,9 de momento, e *mini-batch* de tamanho um. Para cada modelo, é feito o *fine-tuning* usando a mesma função de perda que foi utilizada para o gerar. Isto é, se o modelo foi pré-treinado como um modelo de classificação, faz-se o *fine-tuning* como classificação, e assim por diante. Os experimentos foram realizados utilizando as divisões disponibilizadas de treinamento, validação e teste. Ao fim de cada época, o modelo foi testado no conjunto de validação e a *performance* foi medida com a soma da F1 Ponderada com o Kappa. O treinamento termina quando a *performance* fica abaixo do máximo histórico por três épocas consecutivas. Ao fim, foi selecionado o modelo com a melhor *performance*.

Competição	**C1** .867 – 1.117	**C2** .907 – 1.227	**C3** .732 – .909	**C4** .759 – 1.068
Fine-tuning-regressão	**1.178**	1.035	.846	**1.130**
Fine-tuning-classificação	.985	**1.080**	**.902**	1.023
Fine-tuning-ordinal	.908	.097	.724	.569
BERTimbau	1.132	1.105	.722	1.015

Tabela 1: Comparação, para cada competência, entre a *performance* dos modelos da competição contra os deste estudo (BERTimbau). Em negrito, a melhor *performance* entre os modelos deste estudo. Os valores sublinhados indicam uma *performance* melhor do que a da competição.

Na segunda etapa, se investiga se as *features* curadas podem ser usadas para atingir *performances* competitivas em relação aos modelos neurais.

Na terceira etapa, os modelos neurais são agregados com as *features* e, com isso, é criado um modelo de Aprendizado em Dois Estágios.

Na quarta etapa, se chega a um ponto em que há muitas *features* para poucos exemplos de treinamento. Portanto, foram executados algoritmos de seleção de *features* — testando-se dois, ambos disponibilizados pelo scikit-learn.

Por fim, na quinta etapa, se busca uma melhor *performance* através do uso do XGBoost e mudando a configuração da seleção de *features*.

Para facilitar a compreensão, optou-se por fornecer informações mais detalhadas no início de cada etapa da próxima seção.

4. Experimentos

Nesta seção são apresentados os resultados de cada etapa, conforme descrito ao final da seção anterior. Começa-se com o teste dos corretores neurais, previamente treinados no ENEM. Em seguida, testam-se modelos baseados em *features*. Na terceira etapa, combinam-se os dois anteriores. Na etapa seguinte, realiza-se a seleção de *features*. Por fim, busca-se alcançar o melhor resultado.

4.1. Primeira etapa: Modelos Neurais

A primeira investigação consiste em verificar se o uso de modelos pré-treinados para a avaliação do ENEM conduz a resultados superiores aos apresentados na competição.

É razoável supor que os participantes tenham testado múltiplas configurações e conjuntos de hiperparâmetros. Assim, se o *fine-tuning* simples gerar desempenho superior, infere-se que as tarefas são suficientemente próximas e que é possível reaproveitar o conhecimento. Os resultados estão apresentados na Tabela 1, nas linhas iniciadas por "fine-tuning". A designação "regressão ordinal" foi abreviada para "ordinal" nas tabelas.

Na primeira linha da tabela é mostrado o intervalo de valores das *performances* dos competidores para cada uma das competências (C). Para a C1, cuja definição corresponde à competência homóloga do ENEM, pode-se ver que todos os modelos deste estudo alcançaram *performance* ao menos dentro do intervalo da competição. Em especial, o modelo regressor não apenas foi o melhor entre os deste estudo, como também superou os da competição. Observou-se padrão semelhante na C4, também correspondente à competência do ENEM, em que o melhor modelo foi o regressor, novamente acima dos da competição. No entanto, ao contrário de C1, o modelo ordinal apresentou desempenho inferior aos modelos da competição, mostrando que não conseguiu generalizar e que até pode ter perdido capacidades ao aprender a dar notas do ENEM.

Para as competências C2 e C3, o modelo ordinal também teve desempenho inferior aos da competição — especialmente na C2. Nessas competências, a melhor *performance* se deu com o modelo de classificação, o qual foi apenas competitivo contra os modelos já publicados.

A seguir, foi testado se os modelos que apresentaram melhor *performance* com *fine-tuning* teriam resultados superiores caso os pesos iniciais fossem os do BERTimbau-base original — ou seja, sem o *fine-tuning* com redações do ENEM. O resultado encontra-se na última linha da mesma tabela. Apenas a C2 apresentou desempenho superior quando treinada a partir do modelo original. Enquanto na C1 e na C4 é plausível supor que o pré-treinamento no ENEM contribui para a *performance*, não é trivial compreender por que esse efeito ocorre na C3, mas não na C2. Para aprofundar essa questão, seria necessária uma explicação mais detalhada sobre em que consiste cada nota dessas competências, informação que não foi disponibilizada.

Competição	Fonte	C1 .867 – 1.117	C2 .907 – 1.227	C3 .732 – .909	C4 .759 – 1.068
RL	NILC	1.062	*.515*	.763	1.042
FA	NILC	1.079	*.468*	.842	.903
intervalo FA	NILC	1.008 – 1.152	*.374 – .555*	.775 – .905	.804 – .962
RL	iR4S-C	.964	*.635*	.862	.940
FA	iR4S-C	1.102	*.718*	.857	1.012
intervalo FA	iR4S-C	1.025 – **1.179**	*.647 – .794*	.791 – **.958**	.954 – **1.076**
RL	iR4S-F	1.003	*.674*	.783	.978
FA	iR4S-F	1.089	*.679*	.862	1.004
intervalo FA	iR4S-F	1.017 – 1.161	*.616 – .758*	.793 – .943	.955 – 1.062
RL	Neural	**1.178**	**1.053**	**.919**	**1.109**
FA	Neural	1.085	1.037	.870	1.019
intervalo FA	Neural	1.038 – 1.133	.971 – **1.096**	.819 – .935	.983 – 1.071

Tabela 2: Comparação dos modelos baseados em *features* contra os da competição. O sublinhado indica *performance* melhor do que a da competição. Em itálico, indicam-se *performances* abaixo das da competição. Em negrito, a melhor *performance* dentro do mesmo tipo de classificador por competência.

Esses resultados podem ser considerados relevantes por três motivos principais: (1) embora o modelo ordinal tenha sido considerado o mais eficaz na correção do ENEM em Silveira et al. (2024), não foi capaz, nesse caso, de transferir adequadamente o conhecimento; (2) nas duas competências equivalentes às do ENEM, foi possível superar o estado da arte; (3) em três das quatro competências observou-se benefício decorrente do pré-treinamento, em comparação à simples inicialização. Quanto ao primeiro ponto, vale destacar que as métricas usadas no estudo do ENEM diferem das adotadas aqui, uma vez que a regressão ordinal apresenta maior afinidade com o Kappa Quadrático, ao passo que aqui se recorre ao Kappa de Cohen tradicional.

4.2. Segunda etapa: *features* Curadas

A seguir, serão avaliadas *features* provenientes de três origens distintas: calculadas por programas de análise textual — NILC-Metrix e iRead4Skills (iR4S) — e extraídas dos modelos neurais da etapa anterior. O objetivo consiste em verificar se tais *features* apresentam desempenho competitivo quando empregadas como dados de entrada em algoritmos tradicionais, como Regressão Linear (RL) e regressão por Florestas Aleatórias (FA). Em ambos os casos, a predição é arredondada para a nota válida mais próxima. Os resultados desta etapa são apresentados na Tabela 2. As *performances* superiores às da competição estão destacadas via sublinhado, as inferiores aparecem em itálico, e a melhor *performance* entre os algoritmos aqui apresentados encontra-se em negrito.

A RL é determinística e, portanto, produz sempre ao mesmo resultado, de modo que uma única execução se mostra suficiente. As FA, por sua vez, incorporam um elemento de aleatoriedade entre seus parâmetros, o que desaconselha a utilização de apenas uma execução isolada. Por se tratar de um algoritmo computacionalmente leve, foram realizadas 100 execuções para reduzir o efeito da aleatoriedade. Primeiramente, é reportada a *performance* média e, em seguida, o intervalo de *performance*, isto é, a melhor e pior performance entre as execuções. Este intervalo é particularmente interessante, pois indica que determinado nível de *performance* pode ter sido alcançado apenas por acaso e que, caso esse valor fosse o melhor observado, poderia ter induzido a comunidade a investir em uma linha de investigação fundada em resultados fortuitos.

As *features* neurais são, ao todo, oito: a nota predita pelo Regressor, o softmax dos logits para cada uma das seis classes[4] e qual é a classe prevista.

O NILC-Metrix calcula 72 características e o iR4S, 656, valor relativamente próximo ao do número de instâncias de treinamento. Entre essas *features* encontram-se variações de métricas

[4]Apesar de haver no DEF cinco classes (de 1 a 5), no ENEM existem seis (de 0 a 5), optou-se por apresentar a confiança inteira do modelo.

	features	**tamanho**	**C1**	**C2**	**C3**	**C4**
RL anterior	—	—	.964 – 1.178	.515 – .1053	.763 – .919	.940 – 1.109
RL	completo	736	1.026	.636	.836	.901
FA anterior	—	—	1.079 – 1.102	.468 – 1.037	.842 – .870	.903 – 1.019
FA	completo	736	1.130	1.104	.871	1.039
FA	sqrt	27	1.151	1.015	.949	1.066

Tabela 3: Na primeira divisão: *performance* da RL utilizando todas as *features*, comparada com os intervalos de *performance* dos grupos em separado. Na segunda divisão: semelhante ao anterior, mas utilizando a média de 100 execuções e variando o número de *features* disponíveis.

aplicadas a um mesmo objeto. Por exemplo, ao medir o tamanho do parágrafo, podem ser gerados valores de média, mediana, mínimo, máximo, moda e variância, entre outros. Foi então conduzido um experimento em que tais *features* foram filtradas de modo a reter apenas a média nesses casos, reduzindo o total de *features* para 225. Por esse motivo, distinguem-se o iR4S-C (iRead4Skills-Completo) e o iR4S-F (iRead4Skills-Filtrado).

Começando pelos resultados usando as *features* do NILC, pode-se ver que as *performances* foram competitivas com os modelos mais complexos nas competências C1, C3 e C4. Comparando a RL contra as FA, não há um vencedor claro, pois metade das vezes um foi melhor do que o outro. Apesar disso, a RL apresentou, exceto na última competência, *performances* que ficam dentro do intervalo das FA. Cabe destacar que, na C1, o limite superior de desempenho superou o obtido na competição. Esses resultados mostram que as *features* calculadas pelo NILC são competitivas com os modelos neurais.

Em seguida, pode-se ver que a versão completa das *features* do iR4S seguiu uma lógica semelhante à do NILC: foi competitiva nas competências C1, C3 e C4. Aqui, porém, a *performance* média das FA foi melhor do que a RL em três competências. Além disso, nessas três competências, o intervalo superior das FA foi melhor do que os da competição, indicando também o potencial desse conjunto de *features*. Digno de nota, o intervalo superior das três competências foi o maior valor de intervalo dentro delas. Quando o número de *features* do iR4S é filtrado, o desempenho da RL aumentou em três casos, enquanto o das FA diminuiu noutros três. Ou seja, a *performance* tendeu a aumentar para o caso determinístico e mesmo aqui a *performance* continua competitiva nas três competências elencadas anteriormente.

Por fim, as *features* de origem neural apresentaram os melhores resultados via RL, sendo três delas melhores do que as da competição e, na que não ganhou, foi competitiva com os resultados anteriores. Em todas as competências, o RL foi melhor do que a média das FA.

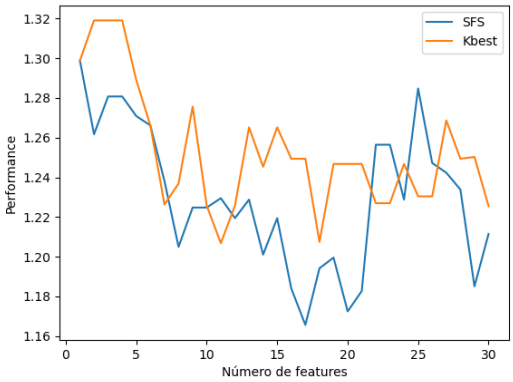
Embora na segunda competência nenhum algoritmo tenha superado o desempenho obtido na competição, conclui-se nessa etapa que as *features* oriundas das três fontes podem ser utilizadas para conseguir *performances* parelhas ou, em alguns casos, superiores às da competição. Na etapa seguinte, serão combinadas todas as fontes.

4.3. Terceira etapa: Aprendizado em Dois Estágios

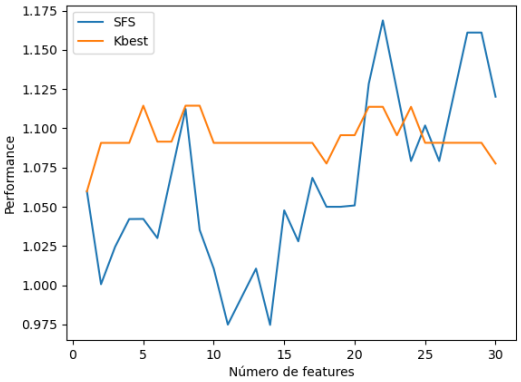
Nesta etapa, foram combinadas as *features* curadas através de um sistema de aprendizado em dois estágios. Para isso, foram concatenadas todas as *features* curadas da etapa anterior, ou seja: (a) todas as *features* do iR4S, (b) todas as *features* do NILC e (c) as 8 *features* neurais. Não foi utilizado o modelo ordinal porque esse não teve a melhor *performance* em nenhum dos casos e, frequentemente, teve *performance* pior do que os da competição. Esse grande conjunto (736) de *features* foi, então, passado para um preditor tradicional.

Como o objetivo ideal era obter um preditor determinístico, o foco inicial recaiu sobre a RL. Os resultados dos experimentos encontram-se na primeira divisão da Tabela 3. Nesse caso, observa-se que as *performances* situaram-se, na maioria das vezes, dentro do intervalo previamente estabelecido e que, especificamente na C4, a combinação de todas as *features* resultou em desempenho inferior. Dessa forma, a utilização dos grupos puros apresentados anteriormente se mostraria sempre mais vantajosa.

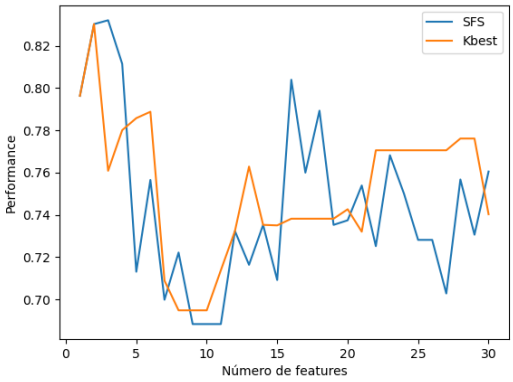
Ainda assim, levanta-se a hipótese de que a combinação não tenha sido proveitosa em razão da quantidade de instâncias. Com esse intuito, foi conduzido um segundo experimento utilizando FA. Nesse caso, explorou-se um de seus hiperparâmetros: o número de *features* consideradas.



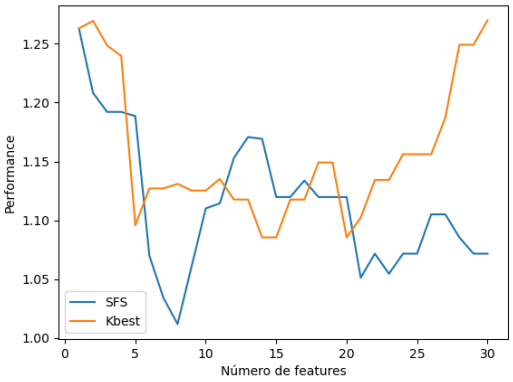
(a) C1: Registro Formal.



(b) C2: Coerência Temática.



(c) C3: Estrutura Retórica Narrativa.



(d) C4: Coesão Textual.

Figura 2: *performance* de um RL com cada número de *feature* escolhido por cada um dos dois algoritmos para cada uma das competências.

Esse valor foi variado entre o padrão (completo) e a raiz quadrada desse valor (*sqrt*). Nesse último caso, 27 *features* são selecionadas aleatoriamente pela biblioteca para executar o algoritmo. Dadas as aleatoriedades da FA, foram realizadas 100 execuções para cada configuração, reportando-se a média das *performances*. Os resultados encontram-se na segunda divisão da Tabela 3.

Nessa divisão observa-se, em primeiro lugar, que as FA apresentaram desempenho consistentemente superior ao da RL. Em segundo lugar, verifica-se que, em comparação com os resultados anteriores das FA, a combinação das *features* produziu melhores resultados. Por fim, em três casos, a redução do número de *features* mostrou-se mais vantajosa.

Até o momento, a RL tem funcionado como um bom *proxy* para o intervalo de *performances* das FA. Com efeito, ao contabilizar quantas vezes a RL se encontra dentro do intervalo das FA na Tabela 2, nos experimentos com NILC, iR4S-C e Neural, observa-se que em 5 casos a RL permaneceu dentro do intervalo, em 4 casos apresentou valores inferiores e em 3 casos valores superiores. Além disso, como a redução do número de *features* resultou em melhores *performances* tanto para a RL na etapa 2 (na comparação entre iR4S-C e iR4S-F) quanto para as FA na etapa atual, considerou-se pertinente realizar um processo de seleção de *features*, que constituirá a próxima etapa.

4.4. Quarta etapa: Seleção de *features*

Nesta etapa foram utilizados algoritmos de seleção de *features* com o objetivo de identificar um subconjunto que proporcione o melhor desempenho. Um efeito colateral desse processo consiste em revelar quais *features* se mostram mais relevantes para a predição da nota.

Para esse fim, foram utilizados dois algoritmos: K-best e SFS — ambos implementados na biblioteca scikit-learn. O primeiro realiza análise univariada, enquanto o segundo busca o melhor subconjunto de modo guloso. O K-best apresenta execução relativamente rápida — pois ele calcula a relação estatística de cada *feature* com o rótulo. O SFS, por sua vez, apresenta maior custo com-

conjunto	preditor	seletor	C1	C2	C3	C4
completo	RL	Kbest	1.196 (02)	1.047 (05)	.929 (2)	1.131 (30)
	RL	SFS	**1.218** (01)	**1.131** (22)	.941 (3)	**1.163** (01)
sem neurais	RL	SFS	1.116 (08)	.683 (14)	.882 (15)	.991 (10)
neurais adicionadas	RL	SFS	1.161 (16)	1.050 (22)	**.959** (23)	1.064 (18)
sem neurais	XG	SFS	1.058 (08)	.741 (14)	.822 (15)	.995 (10)
neurais adicionadas	XG	SFS	1.154 (16)	1.062 (22)	.930 (23)	1.082 (18)

Tabela 4: *Performance* em cada competência do RL usando as *features* selecionadas por cada algoritmo de seleção. Em parênteses, o número de *features* utilizadas. Em negrito, a melhor *performance* na competência respectiva.

putacional, já que, a cada iteração i é necessário treinar e avaliar $features+1-i$ modelos — para descobrir, por exemplo, as três features mais importantes, são necessárias três iterações.

Como o objetivo consiste em otimizar um parâmetro, nesta etapa emprega-se o conjunto de validação para essa busca. Considerando que 3 dos 4 melhores desempenhos na etapa anterior foram obtidos com 27 *features*, a *performance* foi calculada até o limite de 30 *features*. Os resultados na divisão de validação são apresentados na Figura 2.

Os gráficos mostram tendência de queda nas competências C1 e C3, com melhor *performance* nas primeiras iterações. Além disso, na C1 o melhor resultado foi obtido pelo algoritmo K-best, enquanto na C3 tal coube ao algoritmo SFS. Na última competência, C4, o K-best também superou o SFS, encerrando com tendência de alta, ao passo que o SFS manteve tendência de baixa. Por fim, observou-se declínio da C2 nas últimas iterações em ambos os algoritmos. Diante do empate entre eles, não foi possível descartar nenhum, razão pela qual ambos foram testados na etapa seguinte.

Na primeira parte da Tabela 4 apresenta-se a *performance* da RL, no conjunto de teste, considerando as *features* selecionadas por cada um dos métodos de seleção, bem como o número de *features* utilizadas.

Todas as melhores *performances* foram obtidas com o SFS. Em três das quatro *performances*, os resultados alcançados nesta etapa corresponderam às melhores *performances* (determinísticas) até o momento, superando inclusive o estado da arte. Com isso, o estado da arte foi aumentado em 9% para a C1, 3,5% para a C3 e 8,9% para a C4. É interessante observar que esses resultados se aproximam do limite superior registrado na Tabela 3, e que apenas na C4 o valor obtido nesta etapa o supera. Outro aspecto relevante é que a quantidade de *features* selecionadas é bastante reduzida e consiste, na maioria dos casos, em *features* neurais.

A partir desse ponto, levantam-se duas questões: (a) O que ocorreria se a seleção das *features* não neurais fosse realizada primeiro, com a posterior inclusão das neurais? (b) Seria possível utilizar essas *features* como entrada para um algoritmo mais robusto, como o XGBoost? Ambas as questões serão examinadas na etapa seguinte.

4.5. Quinta etapa: Melhor *performance*

Esta etapa investiga as duas questões levantadas no final da etapa anterior.

Para responder à primeira questão, a etapa anterior foi repetida utilizando o seguinte conjunto de *features*: aquelas computadas pelo NILC concatenadas com as do iR4S-C. Esse conjunto totaliza 728 *features*. Assim, o conjunto de validação foi empregado para selecionar um máximo de 30 *features* dentre as 728. Em seguida, foi testado se a adição das *features* neurais ao conjunto selecionado resultaria em melhoria da *performance*, em comparação à seleção realizada diretamente sobre o conjunto completo. Os resultados desse experimento encontram-se na segunda parte da Tabela 4.

A análise da segunda parte da tabela mostra que a inclusão das *features* neurais resultou em melhoria de *performance* em todos os casos. No entanto, ao comparar essa segunda parte com a primeira da tabela, observa-se que, em termos de *performance*, a seleção de *features* realizada diretamente sobre o conjunto completo apresenta resultados superiores.

Para responder à segunda questão levantada no início desta etapa, foi otimizado um XGBoost utilizando os mesmos conjuntos de *features* apresentados na segunda parte da Tabela 4. Os resultados encontram-se na terceira parte da tabela.

A análise desses resultados mostra que a adição das *features* neurais exerceu um efeito positivo sobre a *performance*. Em dois casos (C2 e C4), o XGBoost superou a RL com a mesma quantidade de *features*, embora não tenha ultrapassado os melhores resultados obtidos nas etapas anteriores.

5. Discussão

Nesta seção serão discutidos, de forma geral, os resultados obtidos. Primeiramente, procede-se a uma revisão das cinco etapas apresentadas na seção anterior. Em seguida, são examinadas as *performances*. Por fim, realiza-se uma análise dos erros.

A primeira etapa deste trabalho é comparável às estratégias adotadas durante a competição em que o *dataset* foi disponibilizado. Nessa fase, demonstrou-se que foi possível aproveitar o conhecimento prévio dos modelos treinados para corrigir as competências do ENEM para, agora, corrigir as competências do DEF. Destacam-se alguns aspectos relevantes: (a) no trabalho referente ao ENEM, o modelo ordinal apresentou melhor desempenho (Silveira et al., 2024), enquanto neste estudo apresentou desempenho consistentemente inferior, sem explicação evidente; possivelmente, tal deveu-se à escolha de hiperparâmetros ou ao fato de, no ENEM, a métrica então utilizada ter sido o Kappa Quadrático, mais adequado à regressão ordinal; (b) além de superarem os modelos da competição, os modelos pré-treinados no ENEM também obtiveram desempenho superior ao do modelo original, com exceção da competência C2, cuja avaliação é diretamente afetada pela mudança de gênero das redações. Assim, conclui-se que, nesta etapa, a transferência de conhecimento entre tarefas é viável por meio dos modelos neurais empregados.

Na segunda etapa foram introduzidas *features* curadas provenientes do NILC-Metrix — responsável pelo cálculo de medidas de complexidade e coesão textual —, do iR4S — projeto voltado para a avaliação da inteligibilidade textual — e dos modelos neurais previamente treinados. Essas *features* foram empregadas como entrada em algoritmos tradicionais, como Regressão Linear (RL) e Florestas Aleatórias (FA). Entre as quatro competências, apenas as *features* neurais mostraram-se competitivas na segunda competência, C2. Quando utilizadas em uma RL, alcançaram *performances* superiores às da competição nas competências C1, C3 e C4, o que pode ser interpretado como uma forma simples de agregar os modelos neurais. As demais *features*, não neurais, não obtiveram *performances* determinísticas acima das da competição, mas ainda assim revelaram-se competitivas em três competências. Esse resultado indica que, neste *dataset*, os modelos neurais não são completamente dominantes, sendo, portanto, possível desenvolver modelos mais explicáveis a partir dessas *features*. Além disso, demonstrou-se que, em virtude da aleatoriedade das FA, seria plausível que algum competidor obtivesse vantagem em determinada competência com base em um modelo de *features*, o que poderia ter despertado maior interesse por essa linha de modelos.

Na etapa seguinte, todas as características foram combinadas para formar um modelo de Aprendizado em Dois Estágios. Nesse caso, a *performance* média das FA superou o intervalo obtido pelos grupos separados, resultado que não se verificou para a RL. A hipótese levantada é que, neste ponto, o número de *features* tornou-se muito próximo ao número de exemplos, o que comprometeu a capacidade de generalização dos modelos. Essa hipótese foi testada mediante a redução do número de *features* disponíveis para as FA, o que resultou em desempenhos superiores aos obtidos quando todas as *features* eram consideradas. Assim, concluiu-se que seria necessária uma etapa adicional de seleção de *features*.

Na quarta etapa verificou-se um empate entre os métodos de seleção de *features*, razão pela qual ambos foram utilizados. No conjunto de teste, entretanto, o método SFS apresentou melhor desempenho. Nessa fase, alcançaram-se os maiores resultados obtidos até então em todas as competências, com as competências C1, C3 e C4 superando os valores da competição. Uma característica marcante dessa etapa é que o processo de seleção de *features* realizou uma filtragem bastante rigorosa, selecionando, na maior parte dos casos, apenas *features* neurais.

Na quinta etapa buscou-se investigar se seria possível, nesse ponto, melhorar ainda mais as *performances*. A primeira análise consistiu em realizar a seleção inicial entre as *features* não neurais, o que resultou em conjuntos maiores, porém com *performances* inferiores às obtidas pela seleção completa. Posteriormente, todas as *features* neurais foram adicionadas a esse conjunto, o que proporcionou uma melhoria da *performance*, embora ainda aquém da seleção completa — com exceção da terceira competência, na qual se observou um ligeiro aumento adicional.

Por fim, testaram-se as *features* selecionadas em um XGBoost otimizado, mas os resultados não superaram os anteriores, possivelmente porque a seleção realizada pelo SFS requer um algo-

	C1			C2			C3			C4		
	soma	F1	K	soma	F1	K	soma	F1	K	soma	F1	K
Melhor comp.*	1.117	.708	.414	**1.227**	**.679**	**.548**	.909	.626	.286	1.068	.702	.367
RL-completo-SFS	**1.218**	**.735**	**.482**	1.131	.641	.490	.941	.635	.305	**1.163**	**.725**	**.437**
RL-neurais-adicionadas	1.161	.717	.444	1.050	.603	.447	**.959**	**.641**	**.317**	1.064	.699	.364

Tabela 5: Comparação das melhores *performances* da competição contra as melhores *performances* deste estudo. A melhor *performance* está destacada em negrito. O asterisco indica que cada valor reportado da competição pode vir de um algoritmo diferente.

ritmo de base, tendo sido utilizado um Regressor Linear para esse fim. Assim, seria necessário repetir todo o processo de seleção especificamente orientado ao XGBoost.

Em resumo, a transferência de conhecimento entre tarefas permitiu superar os algoritmos da competição em três das quatro competências. A partir deste ponto, a análise passou a concentrar-se nos resultados numéricos das *performances*. Na Tabela 5 apresentam-se a *performance* somada, a F1 Ponderada e o Kappa dos dois melhores modelos aqui obtidos, em comparação com o melhor valor reportado no artigo da competição — ou seja, a soma pode vir de um algoritmo, enquanto o F1 de um segundo e o Kappa de um terceiro. Pela Tabela 5, observa-se que a abordagem proposta apresentou baixo desempenho na C2 em todas as métricas, o que se explica pelo fato de essa competência exigir a análise do tema — aspecto não capturado por nenhuma das *features* utilizadas. Na C4, o método que inclui as *features* neurais obteve resultados ligeiramente inferiores às melhores *performances*, enquanto a seleção completa apresentou desempenho superior. De modo geral, os resultados obtidos superaram o estado da arte nas métricas referentes às competências C1, C3 e C4.

Com base na categorização clássica dos valores de Kappa (Landis & Koch, 1977), na C4 houve uma evolução de concordância regular (ou branda) para moderada. A F1 Ponderada, por sua vez, apresenta interpretação menos direta; contudo, como os valores obtidos superam 0,6, infere-se que as classes mais frequentes estão sendo corretamente identificadas. A Tabela 6 apresenta a acurácia por classe e por competência para o algoritmo RL-completo-SFS, enquanto a Tabela 7 mostra os resultados correspondentes para o algoritmo RL-neurais-adicionadas — é válido ressaltar que a distribuição é a mesma nas duas tabelas, a acurácia, porém, difere. As matrizes de confusão respectivas encontram-se no Apêndice A.

No caso do RL-completo-SFS, observa-se que o algoritmo raramente atribui a nota máxima, exceto na competência C3, em que a nota não prevista corresponde à mínima. Nota-se ainda que, na maioria dos casos, a acurácia por classe apresenta fraca correlação com a frequência relativa de exemplos daquela classe. Em outras palavras, a classe mais frequente atinge acurácia superior a 80%, enquanto a menos frequente apresenta acurácia nula. O modelo RL-neurais-adicionadas segue a mesma tendência do anterior, embora com *performances* ligeiramente inferiores na maioria dos casos.

nota	C1	C2	C3	C4
1	10 (50%)	112 (83%)	7(00%)	11(36%)
2	54 (75%)	81 (32%)	12(41%)	55(65%)
3	248 (86%)	153 (81%)	52(42%)	256(85%)
4	53 (32%)	19 (05%)	239(82%)	37(35%)
5	4 (00%)	4 (00%)	59(28%)	10(00%)

Tabela 6: Distribuição dos rótulos e acurácia do preditor (entre parênteses) por classe por competência do modelo RL-completo-SFS.

nota	C1	C2	C3	C4
1	10 (60%)	112 (84%)	7(14%)	11(36%)
2	54 (72%)	81 (19%)	12(33%)	55(54%)
3	248 (87%)	153 (81%)	52(46%)	256(86%)
4	53 (20%)	19 (00%)	239(79%)	37(24%)
5	4 (00%)	4 (00%)	59(35%)	10(00%)

Tabela 7: Distribuição dos rótulos e acurácia do preditor (entre parênteses) por classe por competência do modelo RL-neurais-adicionadas.

Dessa análise conclui-se que a principal necessidade, no momento, é aprimorar a classificação das classes sub-representadas ou, alternativamente, investir na construção de *datasets* com número equilibrado de instâncias por classe.

Adicionalmente, foi analisada uma característica específica deste *dataset*: a presença de dois temas, sendo um deles significativamente dominante. Assim, avaliou-se a *performance* dos dois algoritmos por tema. A Tabela 8 apresenta a soma das *performances*, onde $soma_1$ corresponde à *performance* no tema dominante e $soma_2$ ao tema secundário.

Algoritmo	C1		C2		C3		C4	
	$soma_1$	$soma_2$	$soma_1$	$soma_2$	$soma_1$	$soma_2$	$soma_1$	$soma_2$
RL-completo-SFS	1.252	.445	1.159	.487	.925	1.176	1.194	.489
RL-neurais-adicionadas	1.198	.418	1.082	.315	.937	1.362	1.108	.138

Tabela 8: *Performance* dos modelos selecionados para o tema dominante ($soma_1$) e para o outro tema ($soma_2$).

Essa análise mostra que o tema não dominante, na maioria das vezes, apenas reduz as *performances*, uma vez que a nota obtida no tema dominante é superior à do conjunto de teste tradicional. Uma possível linha de investigação consiste em tratar este *dataset* como se contivesse apenas um tema, eliminando as instâncias correspondentes ao não dominante.

Por fim, foram analisadas as *features* não neurais selecionadas pelos métodos em cada competência. As tabelas completas encontram-se no Apêndice B. O foco recai sobre o modelo RL-neurais-adicionadas, uma vez que o outro selecionou, em sua maioria, apenas *features* neurais.

Começando pela competência C1 — bom uso do português —, seis das oito *features* selecionadas provêm do iR4S e estão relacionadas à avaliação das palavras (tokens) do texto: medem o tamanho, a variedade, o nível de sofisticação, a idade média de aquisição, bem como a quantidade de sinais de pontuação utilizados. As duas *features* oriundas do NILC-Metrix, uma delas mede a variedade de classes gramaticais presentes no texto e a outra avalia o grau de ambiguidade dos verbos empregados. De modo geral, essas *features* mostram-se, de fato, bons aproximadores para a competência em questão.

Para a C2 — responsável por avaliar a relevância do texto relativamente ao tema —, qualquer *feature* disponível limita-se a estabelecer correlações espúrias, uma vez que o tema não é considerado por nenhuma das métricas utilizadas. Uma forma de contornar essa limitação seria desenvolver *features ad hoc*, como a similaridade de cosseno entre as representações dos textos (redação e tema) ou ainda uma *feature* que contabilizasse o número de palavras em comum entre ambos.

Para a C3 — estrutura retórica narrativa — é plausível supor a existência de correlações entre *features* de complexidade sintática e lexical e a adequação ao gênero textual. De fato, entre as *features* selecionadas encontram-se: três que medem o tamanho do texto, cinco que avaliam a frequência lexical de diferentes classes gramaticais, uma relacionada à repetição de palavras e outra à repetição de raízes, duas que calculam a quantidade de vizinhos ortográficos, uma que identifica a palavra mais difícil utilizada, outra que contabiliza o número de verbos no presente, e, por fim, uma que mede a repetição de palavras entre sentenças.

Por fim, a C4 — responsável por avaliar a coesão — foi representada por dez *features*, das quais oito provenientes do iR4S e duas do NILC-Metrix. Entre as do iR4S encontram-se: medidas do tamanho das palavras, da diversidade lexical, da frequência dos lemas, da quantidade de vizinhos ortográficos, do número de palavras após o verbo principal da sentença e da proporção de palavras repetidas em diferentes sentenças. Do NILC-Metrix foram selecionadas uma *feature* relacionada à complexidade de classes gramaticais e outra à complexidade sintática. Enquanto as *features* do NILC-Metrix apresentam maior pertinência em relação à competência, a maioria das do iR4S não parece diretamente relacionada, com exceção da repetição de palavras entre sentenças e do número de palavras posteriores ao verbo.

Assim, conclui-se que os resultados obtidos — em comparação com os da competição — mostram-se consistentes. As *features* selecionadas para C1 e C3 revelaram-se adequadas e resultaram em melhores *performances*. Na C4 verificou-se alguma relação, com *performance* equivalente à da competição. Já na C2 não se observou qualquer relação, e a *performance* foi inferior.

6. Conclusão

Correção Automática de Redação é uma das áreas da IA que tem capacidade de impactar positivamente o ensino, tanto do lado do professor quanto do aluno. Pesquisas deste tópico em português europeu são focadas em identificar a dificuldade do texto de acordo com o Quadro Europeu Comum de Referência para as línguas (Council of Europe, 2020), enquanto no português brasileiro existem pesquisas orientadas à correção automática de redações argumentativas de vestibular e redações narrativas para nível fundamental.

Neste trabalho foi investigada a correção de redações narrativas, tema apresentado em uma competição realizada no âmbito do PROPOR 2024. Nessa competição, sete algoritmos foram testados, em geral envolvendo o uso do BERT, seja para predizer a nota, seja para gerar a representação vetorial da redação. No presente estudo, avançou-se além dessa abordagem, propondo-se um modelo de aprendizado em dois estágios.

Esse modelo é constituído por duas etapas: na primeira, realiza-se o *fine-tuning* de um modelo neural; na segunda, utiliza-se sua saída em conjunto com outras *features* para predizer a nota final. Foram empregados dois modelos neurais: um regressor — cuja saída corresponde à nota — e um classificador — cuja saída indica a confiança em cada classe e a classe mais provável. Essas *features* neurais foram concatenadas com aquelas calculadas por dois sistemas distintos: NILC-Metrix e iRead4Skills. Por fim, considerando o grande número de *features* em contraste com a quantidade relativamente reduzida de exemplos de redação, aplicaram-se algoritmos de seleção de *features* com o objetivo de aprimorar a *performance* de um regressor linear.

O processo de desenvolvimento desse algoritmo foi realizado em cinco etapas: (1) treinamento dos modelos neurais; (2) utilização isolada dos grupos de *features*; (3) combinação dos modelos neurais com as demais *features*; (4) treinamento de Regressão Linear a partir de *features* selecionadas; (5) experimentação com diferentes configurações.

Na primeira etapa demonstrou-se ser possível, com modelos neurais, realizar transferência de aprendizado das redações do ENEM para as redações narrativas em duas competências — dada a grande semelhança entre elas. Na segunda etapa, evidenciou-se que o uso de subconjuntos dos dados pode gerar desempenhos superiores aos apresentados na competição. Na etapa seguinte, verificou-se que, com a combinação dos modelos neurais com as demais *features*, era possível obter *performances* ainda melhores em relação às etapas anteriores. Na quarta etapa, empregou-se um regressor linear com *features* selecionadas, alcançando os seguintes resultados para cada competência: C1 - 1.218, C2 - 1.131, C2 - 0.941 e C4 - 1.163. Esses valores representam melhorias de 9% na C1, 3,5% na C3 e 8,9% na C4. Finalmente, na última etapa, ao serem testadas diferentes configurações, atingiu-se o valor de 0.959 na C3, correspondendo a um aumento de 5,5% em relação ao estado da arte atual.

Em seguida, as *performances* foram apresentadas sob diferentes perspectivas. Primeiramente, decompuseram-se os resultados nas métricas constituintes, de modo a possibilitar comparações futuras. Também foi reportada a acurácia por classe, evidenciando que a principal dificuldade reside na predição das classes com poucos exemplos — geralmente correspondentes aos extremos (melhor ou pior nota). Além disso, verificou-se que os modelos, em geral, apresentam melhor *performance* em um dos temas do *dataset*. Por fim, analisaram-se as *features* selecionadas, concluindo-se que as *performances* observadas são consistentes com as características das diversas seleções.

Apesar do aumento nas *performances*, os resultados permanecem aquém do valor de 0,6 de Kappa, usualmente considerado como referência. Nesse sentido, novas investigações podem ser conduzidas a partir de uma seleção mais refinada das *features*, identificando quais são mais adequadas para cada competência. Também é possível incorporar *features ad hoc*, como a similaridade de cosseno entre tema e redação, a contagem de erros ortográficos, entre outras. É possível testar se o pré-treinamento no Essay-br (Marinho et al., 2021) também leva a melhores performances. Por fim, abre-se a possibilidade de explorar a combinação de modelos multilingues e redações em outros idiomas para realizar predições já na primeira etapa. Outro caminho possível consiste em investigar a utilização de outros modelos clássicos e também modelos neurais maiores, baseados em *Decoders* ou modelos baseados em instruções.

Agradecimentos

Este trabalho foi financiado por fundos nacionais portugueses através da Fundação para a Ciência e a Tecnologia, I.P. (FCT) no âmbito dos projetos UID/50021/2025 e UID/PRR/50021/2025 e pela Comissão Europeia (Projeto: iRead4Skills, Número da subvenção: 1010094837, Tópico: HORIZON-CL2-2022-TRANSFORMATIONS-01-07, DOI: 10.3030/101094837). Também foi financiado pela CAPES (88881.982728/2024-01) e FAPESP (2022/02937-9).

Referências

Amorim, Evelin & Adriano Veloso. 2017. A multi-aspect analysis of automatic essay scoring for Brazilian Portuguese. Em *Student Research Workshop at the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 94–102.

Chen, Tianqi & Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. Em *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785—-794. 10.1145/2939672.2939785

Council of Europe. 2020. *Common European framework of reference for languages: Learning, teaching, assessment. companion volume.* Strasbourg: Council of Europe

Hamner, Ben, Jaison Morgan, lynnvandev, Mark Shermis & Tom Vander Ark. 2012. The Hewlett foundation: Automated essay scoring. Kaggle.

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye & Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. Em *Advances in Neural Information Processing Systems*, vol. 30, 3149—3157.

Landis, J Richard & Gary G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics* 33. 159–174. 10.2307/2529310

Leal, Sidney Evaldo, Magali Sanches Duran, Carolina Evaristo Scarton, Nathan Siegle Hartmann & Sandra Maria Aluísio. 2024. NILC-Metrix: Assessing the complexity of written and spoken language in Brazilian Portuguese. *Language Resources and Evaluation* 58(1). 73–110. 10.1007/s10579-023-09693-w

de Lima, Tiago Barbosa, Elyda Freitas & Valmir Macario. 2024. AESVoting: Automatic essay scoring with BERT and voting classifiers. Em *16th Conference on Computational Processing of Portuguese (PROPOR)*, 6–9.

Liu, Jiawei, Yang Xu & Yaguang Zhu. 2019. Automated essay scoring based on two-stage learning.

Marinho, Jeziel, Rafael Anchiêta & Raimundo Moura. 2021. Essay-BR: a Brazilian corpus of essays. Em *III Dataset Showcase Workshop*, 53–64.

Marinho, Jeziel, Rafael Anchiêta & Raimundo Moura. 2022a. Essay-BR: a Brazilian corpus to automatic essay scoring task. *Journal of Information and Data Management* 13(1). 65–76. 10.5753/jidm.2022.2340

Marinho, Jeziel C, Fábio Cordeiro, Rafael Anchiêta & Raimundo S Moura. 2022b. Automated essay scoring: An approach based on ENEM competencies. Em *Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*, 49–60. 10.5753/eniac.2022.227202

Mello, Rafael Ferreira, Hilário Oliveira, Moésio Wenceslau, Hyan Batista, Thiago Cordeiro, Ig Ibert Bittencourt & Seiji Isotanif. 2024. PROPOR'24 competition on automatic essay scoring of Portuguese narrative essays. Em *16th Conference on Computational Processing of Portuguese (PROPOR)*, 1–5.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12. 2825–2830.

Ribeiro, Eugénio, Nuno Mamede & Jorge Baptista. 2024. Exploring the automated scoring of narrative essays in Brazilian Portuguese using transformer models. Em *16th Conference on Computational Processing of Portuguese*, 14–17.

Rodrigues, João, Luís Gomes, João Silva, António Branco, Rodrigo Santos, Henrique Lopes Cardoso & Tomás Osório. 2023. Advancing neural encoding of Portuguese with transformer Albertina PT-*. Em *EPIA Conference on Artificial Intelligence*, 441–453. 10.1007/978-3-031-49008-8_35

Silveira, Igor Cataneo, André Barbosa, Daniel Silva Lopes da Costa & Denis Deratani Mauá. 2025. Investigating universal adversarial attacks against transformers-based automatic essay scoring systems. Em *Brazilian Conference on Intelligent Systems (BRACIS)*, 169–183.

Silveira, Igor Cataneo, André Barbosa & Denis Deratani Mauá. 2024. A new benchmark for automatic essay scoring in Portuguese. Em *16th Conference on Computational Processing of Portuguese (PROPOR)*, 228–237.

de Sousa, Rogério F, Jeziel C Marinho, Francisco A R Neto, Rafael Anchiêta & Raimundo S Moura. 2024. PiLN at PROPOR: A BERT-based strategy for grading narrative essays. Em *16th Conference on Computational Processing of Portuguese (PROPOR)*, 10–13.

Souza, Fábio, Rodrigo Nogueira & Roberto Lotufo. 2020. BERTimbau: pretrained BERT models for Brazilian Portuguese. Em *Brazilian Conference on Intelligent Systems (BRACIS)*, 403–417. 10.1007/978-3-030-61377-8_28

Wilson, Joshua & Amanda Czik. 2016. Automated essay evaluation software in english language arts classrooms: Effects on teacher feedback, student motivation, and writing quality. *Computers & Education* 100. 94–109. 10.1016/j.compedu.2016.05.004

Yannakoudakis, Helen & Ronan Cummins. 2015. Evaluating the performance of automated text scoring systems. Em *10th Workshop on Innovative Use of NLP for Building Educational Applications*, 213–223. 10.3115/v1/W15-0625

A. Matrizes de Confusão

Neste apêndice, apresentam-se na Tabela 9 as matrizes de confusão do modelo RL-completo-SFS, e nas Tabela 10 as correspondentes ao modelo Neurais-Adicionadas.

		referência C1						**referência C2**						**referência C3**						**referência C4**					
		1	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**	
	1	5	3				8	93	12	3	2		110						0	4		1			5
	2	5	41	18			64	18	26	24	3	1	72	5	5	4	3		17	7	36	19	1		63
$\hat{y}$	3		10	214	36	3	263	1	43	125	13	2	184	2	5	22	13	2	44		19	219	23	3	264
	4			16	17	1	34			1	1	1	3		2	25	197	40	264			17	13	7	37
	5						0						0			1	26	17	44						0
		10	54	248	53	4		112	81	153	19	4		7	12	52	239	59		11	55	256	37	10	

Tabela 9: Matriz de Confusão para todas as competências utilizando o preditor RL-completo-SFS

		referência C1						**referência C2**						**referência C3**						**referência C4**					
		1	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**		**1**	**2**	**3**	**4**	**5**	
	1	6	3	1			10	95	14	2	3	1	115	1					1	4	1	1			6
	2	4	39	16			59	17	16	26	2		61	5	4	1	4		14	6	30	14	1		51
$\hat{y}$	3		11	217	42	3	273		51	125	14	3	193		6	24	18	3	51	1	24	222	27	6	280
	4			14	11	1	26						0	1	2	26	190	35	254			19	9	4	32
	5						0						0			1	27	21	49						0
		10	54	248	53	4		112	81	153	19	4		7	12	52	239	59		11	55	256	37	10	

Tabela 10: Matriz de Confusão para todas as competências utilizando o preditor RL-neurais-adicionadas

B. *Features* Selecionadas

Neste apêndice são apresentadas as *features* selecionadas por cada algoritmo. A Tabela 11 contém os resultados correspondentes, enquanto a Tabela 12 apresenta as *features* selecionadas quando o processo considerou todas as disponíveis. Para mais informações sobre as *features*, recomenda-se consultar o NILC-Metrix Leal et al. (2024) e a página do iR4S [5].

As oito *features* neurais possuem a seguinte denominação:

- grade_BERT: a nota predita pelo BERT regressor;
- max_score: a nota predita pelo BERT classificador;
- confidence_X: a confiança do BERT classificador na nota X.

[5] https://gitlab.hlt.inesc-id.pt/iread4skills/docs

	C1	C2	C3	C4
1	'cw_freq_brwac'	'nouns_ambiguity',	'sentences_per_paragraph'	'cw_freq_brwac'
2	'verbs_ambiguity'	'root_ttr'	'word_length_sum'	'yngve'
3	'word_length_avg'	'lex_freq_content_max'	'word_length_q3'	'word_length_90p'
4	'lex_freq_min'	'p99.5-p99.9_freq_lemma_ratio'	'root_ttr'	'word_lemma_length_iqr'
5	'lex_freq_lemma_median'	'ortho_neighbors_avg_freq_dolch'	'lex_freq_min'	'corrected_ttr'
6	'p0-p75_freq_ratio'	'familiarity_skewness'	'lex_freq_lemma_median'	'lex_freq_lemma_median'
7	'age_of_acquisition_kurtosis'	'concrete_ratio'	'lex_freq_content_lemma_min'	'ortho_neighbors_avg_freq_q1'
8	'punctuation_count'	'abstract_ratio'	'lex_freq_content_lemma_q1'	'ortho_neighbors_+freq_std'
9		'fractional_numerals_count'	'lex_freq_function_lemma_dolch'	'words_after_verb_sum'
10		'possessive_pronouns_count'	'p0-p75_freq_ratio'	'shared_word_ratio'
11		'indefinite_pronouns_count'	'ortho_neighbors_avg_freq_q1'	
12		'present_tense_count'	'ortho_neighbors_+freq_rsd'	
13		'masculine_gender_count'	'familiarity_max'	
14		'shared_content_ratio'	'present_tense_count'	
15			'shared_word_ratio'	
	'grade_BERT', 'max_score', 'confidence_0', 'confidence_1', 'confidence_2', 'confidence_3', 'confidence_4' 'confidence_5'			

Tabela 11: Features selecionadas para cada competência no modelo RL-neurais-adicionadas.

	C1	C2	C3	C4
1	grade_BERT	'grade_BERT'	'max_score'	grade_BERT
2		'adjective_ratio'	confidence_2	
3		'negation_ratio'	confidence_5	
4		'verbs_ambiguity'		
5		'positive_words'		
6		'paragraph_count'		
7		'sentence_length_mode'		
8		'word_lemma_length_median'		
9		'word_syllables_median'		
10		'lex_freq_min'		
11		'lex_freq_verbs_rsd'		
12		'lex_freq_adjectives_kurtosis'		
13		'lex_freq_lemma_std'		
14		'lex_freq_nouns_lemma_rsd'		
15		'lex_freq_verbs_lemma_kurtosis'		
16		'p99.5-p99.9_freq_lemma_ratio'		
17		'ortho_neighbors_avg_freq_skewness'		
18		'familiarity_mode'		
19		'coordinating_conjunctions_count'		
20		'words_after_verb_min'		
21		'dialog_quotes'		
22		'shared_noun_ratio'		

Tabela 12: Features selecionadas em cada competência pelo algoritmo RL-completo-SFS.