UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de Presidente Prudente

Rafael Bezerra de Menezes Rodrigues

Qualidade em conjuntos de dados rotulados: uso do BERT para revisão de anotações e aplicação de saliência para a identificação de vieses

Presidente Prudente
2022

Rafael Bezerra de Menezes Rodrigues

Qualidade em conjuntos de dados rotulados: uso do BERT para revisão de anotações e aplicação de saliência para a identificação de vieses

Dissertação apresentada como parte dos requisitos para obtenção do título de Mestre em Ciência da Computação, junto ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências e Tecnologia da Universidade Estadual Paulista "Júlio de Mesquita Filho".

Orientador: Prof. Dr. Danilo Medeiros Eler

Presidente Prudente
2022

R696q

Rodrigues, Rafael Bezerra de Menezes
Qualidade em conjuntos de dados rotulados: uso do BERT para revisão de anotações e aplicação de saliência para a identificação de vieses / Rafael Bezerra de Menezes Rodrigues. -- Presidente Prudente, 2022
128 p.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências e Tecnologia, Presidente Prudente
Orientador: Danilo Medeiros Eler

1. Ciência da computação. 2. Processamento de textos (Computação). 3. Inteligência artificial. 4. Redes neurais (Computação). I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências e Tecnologia, Presidente Prudente. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA

Câmpus de Presidente Prudente

CERTIFICADO DE APROVAÇÃO

TÍTULO DA DISSERTAÇÃO: Qualidade em conjuntos de dados rotulados: uso do BERT para revisão de anotações e aplicação de saliência para a identificação de vieses

AUTOR: RAFAEL BEZERRA DE MENEZES RODRIGUES
ORIENTADOR: DANILO MEDEIROS ELER

Aprovado como parte das exigências para obtenção do Título de Mestre em Ciência da Computação, área: Computação Aplicada pela Comissão Examinadora:

Prof. Dr. DANILO MEDEIROS ELER (Participaçao Virtual)
Departamento de Matemática e Computação / Faculdade de Ciências e Tecnologia de Presidente Prudente

Profa. Dra. AUREA SORIANO VARGAS (Participaçao Virtual)
Instituto de Computação / Universidade Estadual de Campinas (UNICAMP)

Prof. Dr. DANIEL CARLOS GUIMARÃES PEDRONETTE (Participaçao Virtual)
IGCE / UNESP/Rio Claro (SP)

Presidente Prudente, 01 de dezembro de 2022

Prof. Dr. Danilo Medeiros Eler
Coordenador do Programa

À minha filha Ágata, a luz da minha vida

Agradecimentos

Agradeço ao professor Danilo Eler por ter me acompanhado desde a iniciação científica durante a graduação, até a finalização do mestrado como meu orientador. Obrigado por sempre me dar a liberdade de escolher o meu caminho, garantindo que eu estivesse na trilha certa para cumprir os meus objetivos. Agradeço também aos professores Ivan Rizzo Guilherme e Daniel Pedronette pela oportunidade de trabalhar com inteligência artificial aplicada a processamento de textos na Petrobras. Tem sido um grande desafio, porém enriquecedor e muito empolgante pelos resultados alcançados até então e oportunidades de crescimento.

Agradeço aos meus irmão mais velhos: Matheus, que sem querer me trouxe para o mundo da computação; e Serginho, que sempre zelou por mim e por toda a família. Vocês sempre foram uma inspiração do que eu podia alcançar na vida, e conselheiros que, apesar de eu nem sempre concordar, ouvi e absorvi o que pude.

Agradeço também à minha esposa, Lilian, pela compreensão por todas as horas dedicadas ao estudo e paciência pelos momentos em que não pude estar com você para seguir os meus objetivos.

Por fim, agradeço aos principais responsáveis por eu estar aqui hoje e por ser quem eu sou: meus pais, Ana Maria e Sergio. Agradeço por terem abdicado de muitos luxos para dar a melhor educação possível à mim e aos meus irmãos; não somente pelas escolas boas, mas pela presença, o interesse, e todo o tempo dedicado à nós. Vocês são a minha referência em tudo, verdadeiros exemplos de vida. Agradeço por apoiarem todas as decisões que tomei e que me trouxeram até aqui hoje, e por estarem sempre de prontidão e de braços abertos quando eu preciso.

"A filosofia encontra-se escrita neste grande livro que continuamente se abre perante nossos olhos (isto é, o universo), que não se pode compreender antes de entender a língua e conhecer os caracteres com os quais está escrito. Ele está escrito em língua matemática, os caracteres são triângulos, circunferências e outras figuras geométricas, sem cujos meios é impossível entender humanamente as palavras; sem eles, vagamos perdidos dentro de um obscuro labirinto." – Galileu Galilei

Resumo

A arquitetura Transformer revolucionou a área de processamento de linguagem natural, permitindo a criação do BERT, uma rede neural profunda que, quando lançada, superou o estado da arte em diversas tarefas, como a classificação de textos. No entanto, esta alta performance é acompanhada pela falta de interpretabilidade: o processo de tomada de decisão do BERT é tido como uma caixa-preta, ou seja, é difícil explicar o porquê de uma determinada classificação, com base nas características da entrada e no mecanismo interno do modelo. Dessa forma, torna-se importante o desenvolvimento de técnicas que auxiliem na compreensão do seu funcionamento. A área de XAI (eXplainable Artificial Intelligence) engloba o desenvolvimento dessas técnicas de compreensão, buscando aumentar a confiança dos usuários que utilizam a inteligência artificial, além de entender o que os modelos aprendem e como esse conhecimento é armazenado e utilizado. O presente trabalho descreve técnicas existentes para a compreensão das decisões tomadas pelo BERT, e descreve a aplicação de uma delas para estudo do overfitting e identificação dos vieses aprendidos pelo modelo. Uma mudança na estratégia de treinamento, visando à mitigação dos vieses identificados, levou a uma redução da taxa de falsos positivos em todos os casos observados, mostrando a eficácia da visualização empregada. Outro aspecto importante da classificação de textos, em modelos treinados por aprendizado supervisionado, é a qualidade dos rótulos atribuídos às instâncias do conjunto de treinamento. O presente trabalho também apresenta uma ferramenta para visualização de datasets apresentados de forma compacta e interativa, chamada de Mapa de Instâncias, que auxilia na tarefa de revisão das anotações de conjuntos de dados. Além de permitir a rápida identificação de textos mal rotulados e dos problemas mais críticos de classificação, um experimento mostrou que a combinação da ferramenta com um método de ordenação das instâncias, guiado por um BERT treinado, foi capaz de identificar o dobro de casos mal rotulados quando comparada a uma seleção aleatória dos casos, indicando a sua utilidade para a melhoria de qualidade da anotação dos datasets.

Palavras-chave: XAI, BERT, Transformer, Revisão de anotações.

Abstract

The Transformer architecture revolutionized the natural language processing field, allowing the creation of BERT, a deep neural network that became the state-of-the-art in many tasks, such as text classification. However, its high performance comes with lack of interpretability: the decision-making process of BERT is considered to be a black-box, i. e., it is hard to explain the reason of a specific classification based on the input's characteristics and the internal mechanisms of the model. Therefore, it becomes important to develop techniques that aid in the comprehension of its inner workings. The XAI field (eXplainable Artificial Intelligence) includes the development of such techniques, aiming at gaining the trust of AI's users, besides understanding what the models learn and this knowledge is stored and used. This current work describes existing techniques on the interpretability of BERT's decisions, and describes an application of one of these techniques to the study of model overfitting and the identification of biases learned by the model. A change of the training strategy, aiming at the mitigation of such biases, led to a decrease of false positive rates in all observed cases, showing the efficacy of the employed visualization. Another important aspect in text classification, related to models obtained by supervised training, is the quality of dataset's labels. This current work also presents a tool for the visualization of datasets presented in a compact an interactive way, called Instances Map, which helps on the task of label reviewing. Besides allowing a quick identification of mislabeled instances and the most critical classification errors, an experiment showed that the combination of the tool together with a method of sorting the instances, guided by a trained BERT, was able to identify two times more mislabeled cases, when compared to random selection, pointing to its utility to the improvement of label quality in labeled datasets.

Keywords: XAI, BERT, Transformer, Label reviewing.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

1.1 Contextualização

Explicar o processo de inferência em modelos de processamento de linguagem natural (NLP, na sigla em inglês) tornou-se uma tarefa árdua nos anos recentes. Historicamente estes modelos eram baseados em regras gramaticais e árvores de decisão confeccionadas por especialistas, e portanto inerentemente explicáveis. No entanto, hoje, o estado da arte em diversas tarefas textuais, como classificação de textos e tradução automática, é composto por modelos baseados em redes neurais profundas, cujo processo de tomada de decisão é muito difícil de explicar, não sendo totalmente compreendido (DANILEVSKY et al., 2020).

A última grande revolução na área de NLP foi a criação do Transformer (VASWANI et al., 2017), uma arquitetura de redes neurais baseada totalmente em mecanismos de atenção, que superou a capacidade das redes neurais recorrentes na maioria das tarefas. Desta arquitetura surgiu o BERT (Bidirectional Encoder Representations from Transformers), rede neural profunda objeto de estudo do presente trabalho.

As redes neurais são modelos inspirados no funcionamento do sistema nervoso biológico, que por meio da composição de funções não-lineares, são capazes de aproximar virtualmente qualquer função (GOODFELLOW; BENGIO; COURVILLE, 2016). Seu grande trunfo é a capacidade de extrair características e gerar representações, que são processadas ao longo de suas camadas tornando-se cada vez mais complexas (ZEILER; FERGUS, 2014). No entanto, a grande quantidade de operações, e a profundidade da rede (alto número de camadas de processamento), tornam muito difícil sua interpretação.

A área de XAI (*eXplainable Artificial Intelligence*) consiste no desenvolvimento de técnicas para explicar o funcionamento de modelos de inteligência artificial. Entre os objetivos da área estão o aumento da confiança dos usuários destes modelos; a descoberta de novos conhecimentos que podem estar escondidos em seu funcionamento interno; e a identificação de direções para a melhoria dos modelos, por meio da compreensão de suas limitações (DANILEVSKY et al., 2020). A aplicação de tais técnicas ajuda a explicar, por exemplo, os erros cometidos por um modelo de classificação.

Uma das causas dos erros de classificação, no caso de redes neurais treinadas por aprendizado supervisionado, são os datasets mal rotulados. Neste tipo de aprendizado são apresentados diversos exemplos para a rede, com uma saída associada a cada entrada, e então os parâmetros do modelo são ajustados de acordo com o erro entre valor da saída gerada e o valor esperado. Na tarefa de classificação de textos tem-se rótulos associados a textos. Uma vez que o modelo matemático obtido depende diretamente dos rótulos, a qualidade destas

anotações é muito importante para garantir um bom aprendizado.

1.2 Motivação e Justificativa

Entender os motivos de uma classificação realizada pelo BERT atualmente é uma questão sem respostas satisfatórias, pois nenhuma das técnicas existentes é capaz de elucidar completamente o conhecimento adquirido pelo modelo durante o treinamento, e como ele é utilizado. Por isso, torna-se interessante avaliar o uso de tais técnicas, verificando suas capacidades. Aliada à interpretabilidade do modelo, é possível identificar a origem de erros de classificação no dataset de treinamento, identificando estratégias para melhorar o treinamento e também corrigir suas anotações.

Também é comum que os datasets de treinamento contenham erros de rotulação, o que atrapalha a obtenção de bons modelos. Por isso é importante o desenvolvimento de técnicas e ferramentas que auxiliem na manutenção da qualidade dos dados de treinamento. O uso da IA para guiar a seleção de casos a serem revisados pode ajudar na redução do tempo necessário para a tarefa de revisão, aumentando o custo/benefício deste árduo trabalho.

1.3 Objetivos

1.3.1 Objetivo Geral

O objetivo do projeto é a aplicação de técnicas capazes de auxiliar na tarefa de revisão de anotações guiada por modelos de aprendizado de máquina, assim como a avaliação destes modelos, em particular o BERT treinado para a classificação de textos. Para atingir este objetivo, efetua-se o estudo das técnicas de interpretabilidade existentes, juntamente com sua aplicação para entender os erros de classificação, assim como melhorar a qualidade dos rótulos do dataset.

1.3.2 Objetivos Específicos

Os objetivos específicos do projeto são: i) disponibilização de uma ferramenta para ajudar na exploração de um dataset apresentado de forma compacta, chamada de Mapa de Instâncias; ii) avaliar o uso desta ferramenta no processo de revisão das anotações de um dataset; iii) avaliar o uso do BERT como modelo auxiliar para a revisão das anotações de um dataset; vi) estudo das diferentes técnicas existentes a respeito da interpretabilidade do BERT; v) uso destas técnicas para encontrar fontes de erros no conjunto de treinamento por meio de análise dos efeitos do overfitting; vi) utilizá-las também para avaliar o modelo após a correção desses erros.

1.4 Organização

A Seção 2 contém a fundamentação do trabalho, começando com a apresentação do perceptron, um classificador linear binário inspirado no funcionamento do sistema nervoso, que é a base das redes neurais profundas, partindo para outros conceitos de aprendizado profundo (*deep learning*). Então conceitos do processamento de linguagem natural utilizando redes neurais são apresentados: a arquitetura Word2Vec, que revolucionou o aprendizado de representações semânticas para palavras; o conceito de *transfer learning*; as redes neurais recorrentes que trabalham com dados sequenciais; os mecanismos de atenção; e por fim o Transformer e o BERT. A Seção ainda conta com conceitos de XAI, falando também sobre a interdisciplinaridade da área.

A Seção 3 apresenta as técnicas estudadas, que podem ser divididas em dois grupos: técnicas independentes do modelo (porém aplicadas ao BERT), e técnicas desenvolvidas especificamente para o BERT. Estas últimas ainda dividem-se em visualizações do mecanismo de atenção do Transformer, uma peça chave de seu funcionamento, e visualizações das representações internas.

Depois segue a Seção 4 com a apresentação do Mapa de Instâncias, uma ferramenta desenvolvida para a visualização de datasets inteiros de forma compacta. Esta ferramenta, em conjunto com as predições de um BERT treinado, mostrou-se extremamente eficaz para a identificação de erros de anotação, que devem ser corrigidos para melhorar a qualidade do dataset, e identificação de textos ruins, que atrapalham o aprendizado dos modelos.

Outra contribuição do presente trabalho é descrita na Seção 5, um capítulo onde uma das técnicas de interpretabilidade é utilizada para identificar os vieses aprendidos pelo modelo, ajudando a entender também os efeitos do overfitting. Por fim, apresentam-se as conclusões e trabalhos futuros na Seção 6.

2 Fundamentação

2.1 Redes Neurais

2.1.1 Início: o Perceptron

A história do *Deep Learning* começa com o desenvolvimento de modelos representativos do sistema nervoso. Em 1943, McCulloch e Pitts apresentaram um modelo computacional para imitar o neurônio (MCCULLOCH; PITTS, 1943), demonstrando que redes de neurônios são capazes de expressar diversas proposições lógicas. Este modelo assume que cada neurônio possui um número fixo de sinapses que devem ser excitadas para que ele propague um sinal, e que ainda existem sinapses inibitórias, que ao superarem as sinapses excitatórias por um determinado limiar, não permitem a propagação de nenhum sinal pelo neurônio. Outros aspectos presentes na teoria são a emissão de sinais do tipo "tudo ou nada", que permitem tratar o sinal de saída em cada sinapse como um valor binário, assim como a presença de círculos na rede, permitindo caminhos cíclicos, os quais têm o potencial de indefinidamente manter ativo um neurônio que faça referência a um tempo passado. Ou seja, os círculos permitem que a rede de neurônios possua uma espécie de memória.

Seguindo a ideia conexionista, de que o sistema nervoso armazena informações sobre experiências passadas na forma de conexões entre neurônios, e de que tais informações são recuperadas pelo acionamento destas conexões ao nos depararmos com estímulos similares aos que foram registrados, Rosenblatt apresentou o perceptron, um sistema nervoso hipotético (ROSENBLATT, 1958). Para formular seu modelo, Rosenblatt levou em conta algumas suposições sobre os sistemas nervosos biológicos. As conexões existentes no sistema nervoso são aleatórias no nascimento do organismo, e sua plasticidade permite a alteração destas conexões de maneira duradoura. Estas alterações são resultado de uma grande quantidade de estímulos, onde estímulos similares tendem a percorrer caminhos em direção às mesmas células nervosas, e também podem ser influenciadas por reforços positivos ou negativos. Devido a estas capacidades, o sistema consegue perceber o mundo conforme uma ampla gama de classes de "coisas" distintas.

A Figura 1 apresenta o modelo de um neurônio (Russell e Norvig, 2013), cuja entrada é composta por n valores, representados por x_i, que podem ser a saída de outros neurônios ou características do vetor de entrada. O sinal advindo de cada sinapse - ou valor da entrada - é ponderado por um peso aprendido pela rede, juntamente com o viés do neurônio, representado pela letra "b" (*bias*). Todas as entradas são somadas, e o resultado passa por uma função de ativação, cujo objetivo é determinar se o neurônio irá ou não disparar um sinal, que servirá como entrada para outros neurônios ou como saída para uma classificação binária. Apesar de

simples, este neurônio é capaz de distinguir corretamente entre quaisquer dois grupos de dados linearmente separáveis.

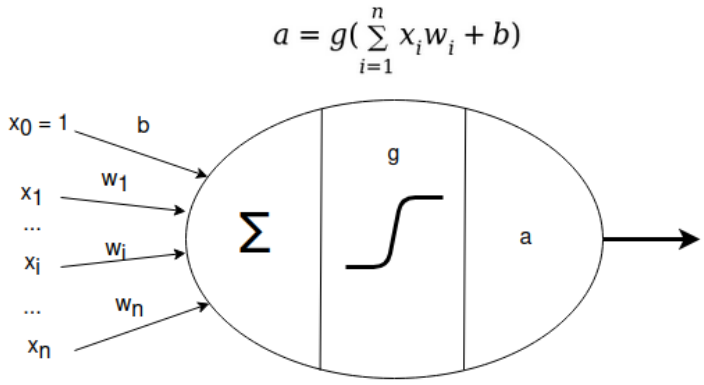


Figura 1 – Modelo computacional de um neurônio - Figura adaptada de (RUSSELL STUART J,)

O treinamento do *perceptron* ocorre por meio da observação de exemplos, onde cada observação resulta na aplicação da Equação 1. Cada peso do neurônio é ajustado de acordo com o erro $y - a_{(x)}$, onde "y" é o valor esperado, que pode ser 0 ou 1 caso a função de ativação gere valores nesta faixa, como a função sigmoide. O erro é então multiplicado pelo valor da entrada relativo àquela conexão, e também por uma taxa de aprendizado α.

$$w_i \leftarrow w_i + \alpha(y - a_{(x)}) * x_i \tag{1}$$

2.1.2 Multi Layer Perceptron

O Multi Layer Perceptron (MLP) é uma rede neural composta por diversas camadas de neurônios, sendo uma camada de entrada, uma de saída, e uma ou mais camadas ocultas. Em comparação ao *perceptron*, que é capaz de resolver apenas problemas com dados linearmente separáveis, o MLP consegue efetuar tarefas muito mais complexas, teoricamente sendo capaz de aproximar qualquer função quando possui duas camadas ocultas, desde que o número de neurônios em cada camada seja suficientemente grande (GOODFELLOW; BENGIO; COURVILLE, 2016).

Cada camada do MLP é chamada de "totalmente conectada" ou "densa", pois todos os neurônios de uma camada estão conectados a todos os neurônios das camadas adjacentes. A Figura 2 ilustra um MLP com duas camadas ocultas.

Seja $W^1 \in \mathbb{R}^{ent} * \mathbb{R}^{oc}$ os pesos que conectam os neurônios da camada de entrada aos neurônios da primeira camada oculta, que podem ser representados por uma matriz, onde "ent" é a quantidade de valores da entrada e "oc" o número de neurônios na camada oculta. E ainda, seja $x \in \mathbb{R}^{ent}$ os valores da entrada, e $b^1 \in \mathbb{R}^{oc}$ o viés dos neurônios da primeira camada oculta.

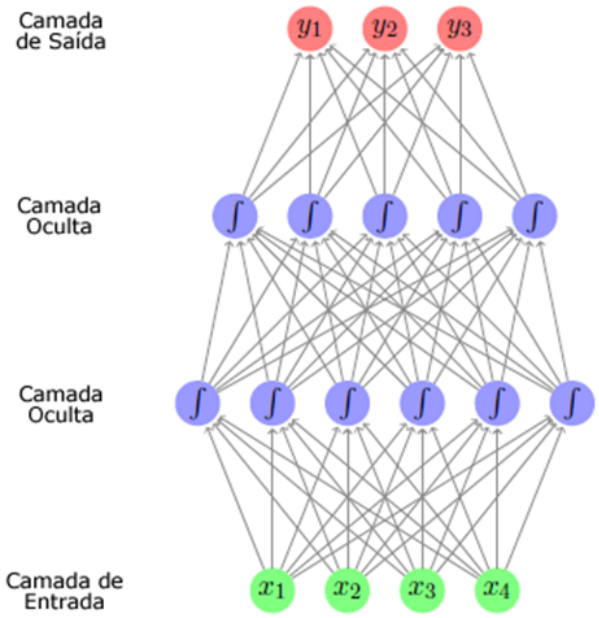


Figura 2 – Multi Layer Perceptron com duas camadas ocultas. Cada flecha representa a conexão entre dois neurônios. A força de cada conexão é representada por um peso, um valor real - Figura traduzida de (GOLDBERG, 2016)

A saída da primeira camada oculta pode ser encontrada por uma simples multiplicação entre um vetor e uma matriz, seguida por uma soma de vetores, como na Equação 2. $y^1 \in \mathbb{R}^{oc}$.

$$y^1 = xW^1 + b^1 \tag{2}$$

Antes de passar os valores para a segunda camada, é aplicada uma função não-linear no valor resultante em cada neurônio. Seja g uma função não-linear, aplicada elemento a elemento no vetor resultante, os valores da segunda camada podem ser encontrados como segue na Equação 3.

$$y^2 = g(y^1) * W^2 + b^2 = g(xW^1 + b^1) * W^2 + b^2 \tag{3}$$

Pode-se estender essa lógica de multiplicação, soma e aplicação da não-linearidades para quantas camadas se desejar. O número de neurônios na camada de saída depende do problema a ser resolvido. Com apenas um neurônio, pode-se resolver problemas de regressão visando um valor real como resultado, ou então efetuar uma classificação binária, que também pode ser modelada com dois neurônios na saída. Já camadas de saída com mais de dois neurônios são utilizadas para problemas com múltiplas classes, onde cada neurônio representa uma classe.

2.1.3 Deep Learning

O aprendizado de máquina (*machine learning*) consiste no desenvolvimento de sistemas de inteligência artificial capazes de aprender padrões a partir de dados. *Deep Learning*

(Aprendizado Profundo) é uma sub-área do aprendizado de máquina, composta por diferentes arquiteturas de redes neurais com muitas camadas, motivo pelo qual utiliza-se o termo "profundo". Um dos grandes diferenciais entre *deep learning* e o *machine learning* tradicional é sua capacidade de aprender representações (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para executar o *machine learning* tradicional é necessário que alguém com conhecimento do domínio prepare os dados, extraindo características que os modelos possam observar para desempenhar suas tarefas. Por exemplo, para efetuar a recomendação de uma cesária por meio da análise de uma ressonância magnética, o médico precisaria saber todas as características relevantes para a decisão, identificá-las e medí-las na imagem, para então decidir como informá-las para o modelo. Já com o *deep learning* os modelos são capazes de analisar diretamente a imagem, pois aprendem a extrair as características e combiná-las em suas diferentes camadas formando novas representações e identificando padrões cada vez mais complexos, até tomar sua decisão final. Ou seja, dispensa o trabalho manual de extração de características por um especialista.

A capacidade de extração de características das redes neurais profundas está exemplificada na Figura 3, onde são exibidos os padrões mais ativados em quatro filtros de uma rede neural convolucional (CNN, da sigla em inglês). Apresenta-se uma das primeiras técnicas para interpretabilidade de CNNs, a deconvolução (ZEILER; FERGUS, 2014), que consiste no processo inverso da convolução. Para cada filtro mostra-se os nove exemplos que mais o ativaram, ou seja, que mais casaram com o padrão procurado pelo filtro. Na parte superior é exibida a porção de cada imagem que mais ativou o filtro, enquanto na parte inferior exibe-se o resultado da técnica, que busca identificar qual foi o padrão da imagem que mais ativou o filtro. Observa-se que os filtros da segunda camada convolucional (parte esquerda) capturam padrões mais simples, como padrões de cores ou círculos. Já a quarta camada (à direita) identifica padrões mais complexos, como água ou ondulações na água, e ainda patas de animais.

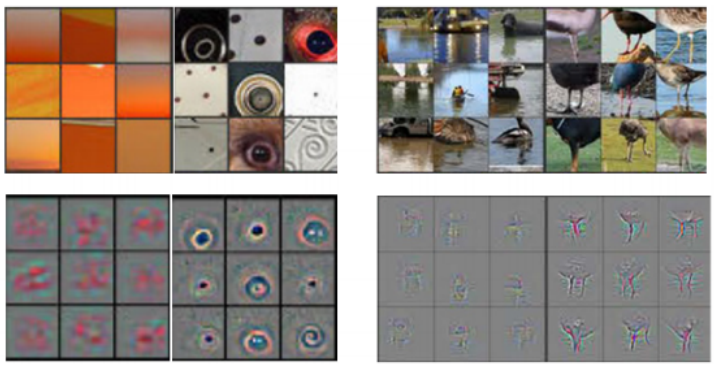


Figura 3 – Características identificadas por diferentes filtros de uma rede neural convolucional. À esquerda dois filtros da segunda camada, e à direita dois filtros da quarta camada - Figura adaptada de (ZEILER; FERGUS, 2014)

Boom do Deep Learning

Dois eventos marcaram a história recente do *Deep Learning*, trazendo um grande interesse pelo assunto: a arquitetura AlexNet em 2012 (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), e o Word2Vec em 2013 (MIKOLOV et al., 2013). A Figura 4 mostra a tendência de busca pelos termos "Deep Learning" no Google, onde é notável a ascensão no interesse pelo assunto a partir de 2013.

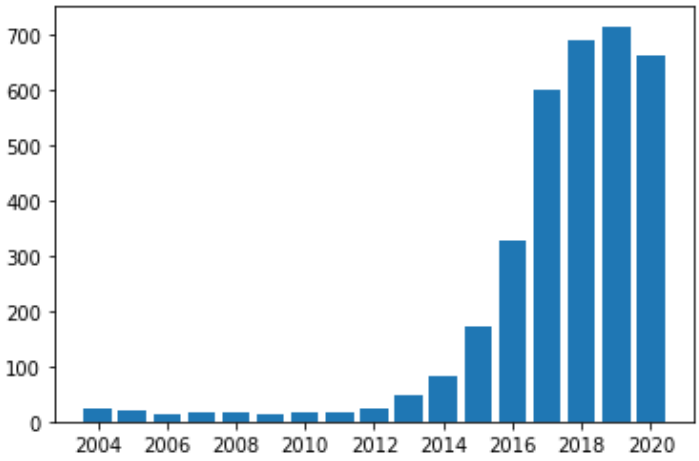


Figura 4 – Tendência de busca por "Deep Learning" no Google

A AlexNet foi um grande marco na área de visão computacional, tratando-se de uma rede neural convolucional com oito camadas: cinco camadas convolucionais e três totalmente conectadas. O funcionamento das camadas convolucionais está fora do escopo deste documento, mas de maneira simplificada, elas consistem em filtros que efetuam uma varredura (convolução) sobre a entrada, identificando as regiões onde determinadas características aparecem. A Figura 5 mostra a arquitetura da AlexNet.

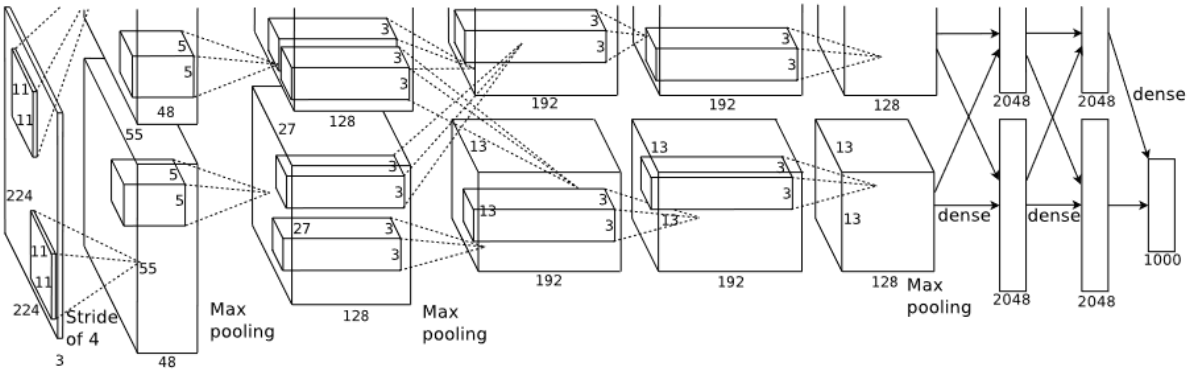


Figura 5 – Arquitetura da AlexNet - Figura retirada de (KRIZHEVSKY; SUTSKEVER; HINTON, 2012)

Essa arquitetura ficou conhecida ao ganhar a competição ImageNet Large Scale Visual Recognition Challenge (ILSVRC) em 2012, ao diminuir em 10 pontos percentuais o erro obtido na classificação de imagens, quando comparada a redes neurais rasas e modelos baseados em extrações de características definidas por humanos. A Figura 6 mostra os resultados da competição entre 2010 e 2015, da mais recente para a mais antiga. De 2012 em diante, todos

os modelos vencedores são redes neurais profundas, chegando a uma rede de 152 camadas em 2015.

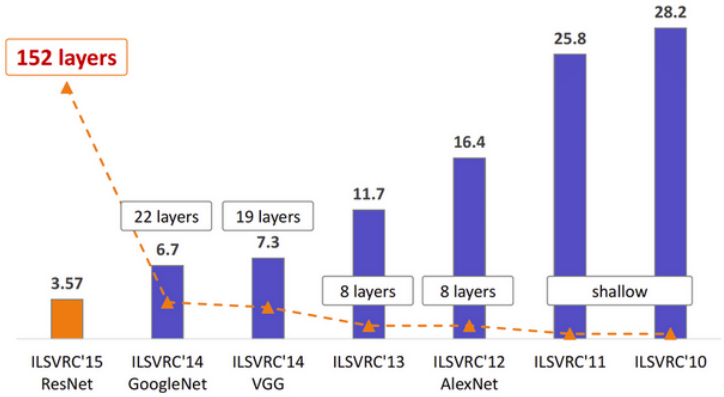


Figura 6 – Erro de classificação da competição ILSVRC - Figura retirada de (NGUYEN et al., 2017)

Já o Word2Vec (MIKOLOV et al., 2013) é uma arquitetura que teve grande impacto na área de processamento de linguagem natural, permitindo a obtenção de excelentes representações semânticas para palavras a partir de grandes quantidades de textos não rotulados. Estas representações então servem como inicialização de vetores de palavras em modelos para diversas tarefas. Ou seja, a arquitetura Word2Vec em si não é feita para a tarefa-fim, mas é utilizada como um meio para a obtenção de representações de palavras que serão utilizadas por outros modelos.

O Word2Vec apresenta duas variantes: Continuous Bag of Word (CBOW), e Skip-Gram (SG). A CBOW prevê uma palavra dado o seu contexto, enquanto a SG prevê o contexto dada uma palavra. O contexto são as palavras ao redor de uma palavra central. Nesta Seção será apresentado o Word2Vec CBOW. Equações de treinamento e métodos de otimização foram omitidos.

Seja s uma sentença com n palavras:

$$s = (x_1, x_2, x_3, ..., x_{n-1}, x_n)$$

Onde cada palavra é representada por um vetor de d dimensões.

Na Figura 7 está representada a arquitetura Word2Vec CBOW, com uma janela de contexto $C = 2$, ou seja, o contexto de uma palavra é composto pelas duas palavras que estão à esquerda e as duas que estão à direita, totalizando uma entrada com quatro palavras. A saída é uma distribuição de probabilidades, onde a palavra central deve ser predita. A arquitetura é composta por duas matrizes de pesos: uma de tamanho V x d, onde V é o tamanho do vocabulário, que é utilizada para representar cada palavra da entrada; e outra de tamanho d x V, que conecta a camada oculta à saída.

Primeiramente é feita uma média das representações das palavras que estão ao redor da palavra alvo. Esta operação dá nome à variante do modelo, Continuous Bag-Of-Words. A

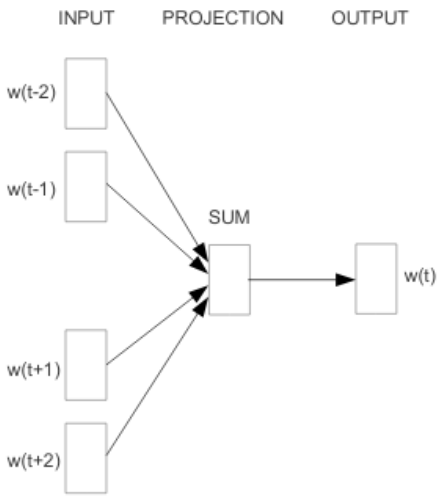


Figura 7 – Ilustração do funcionamento do Word2Vec CBOW - Figura retirada de (MIKOLOV et al., 2013)

camada oculta então processa essa representação do contexto, gerando uma probabilidade para cada palavra do vocabulário. Se a maior probabilidade for correspondente à palavra central, a rede está certa, se for outra palavra, os pesos da rede neural são ajustados.

Durante o treinamento, textos livres são utilizados para extrair uma palavra central e quatro palavras ao seu redor. Ao final do treinamento têm-se uma representação semântica para cada palavra, que é formada ou pela concatenação da representação de cada palavra nas duas matrizes de pesos, ou então pela soma das mesmas.

De acordo com Goldberg (2016), o tamanho da janela de contexto define a relação que se está buscando entre as palavras. Janelas grandes tornam similares termos em um mesmo tópico, como "cachorro", "latido" e "coleira", enquanto janelas pequenas tornam similares palavras que apresentam similaridades funcionais, como "Poodle" e "Pitbull", ou "correndo" e "andando".

2.1.4 Treinamento de Redes Neurais para Classificação

O objetivo do treinamento de uma rede neural é minimizar o erro obtido ao efetuar uma predição, quando comparada ao valor esperado (Goldberg 2016). Para interpretar o erro em um cenário probabilístico, que é o caso da classificação, onde atribui-se uma probabilidade para cada classe conhecida, calcula-se o erro de entropia cruzada categórica (*categorical cross-entropy loss*). Para aplicar este cálculo do erro, a saída da rede neural precisa ser uma distribuição de probabilidades. Ou seja, a soma do *score* dado para todas as classes deve ser igual a um. Para atingir este objetivo, aplica-se a transformação *softmax* à saída da última camada da rede.

Geralmente a rede não é ajustada a cada exemplo visto, mas sim após um pequeno grupo de exemplos, onde toma-se a média do erro entre os exemplos como direção para o

aprimoramento da rede. Após a estimação dos erros para os dados de treinamento, a rede neural é ajustada utilizando um método baseado em gradiente. Calcula-se o gradiente em relação ao erro, e então ajustam-se os parâmetros da rede na direção do gradiente, de tal forma que ela chegue um pouco mais perto da resposta esperada.

Softmax

A *softmax* é uma função de transformação de saída, que modifica todos os valores de um vetor de tal forma que o vetor resultante possui apenas valores positivos e cuja soma é um. Seja $x \in \mathbb{R}^k$ um vetor de números reais com k dimensões, a função *softmax*, aplicada em cada elemento é dada pela Equação 4.

$$softmax(x_i) = \frac{e^{(x_i)}}{\sum_{j=1}^{k} e^{(x_j)}} \tag{4}$$

Segue na Figura 8 um exemplo da aplicação do *softmax* sobre um vetor x.

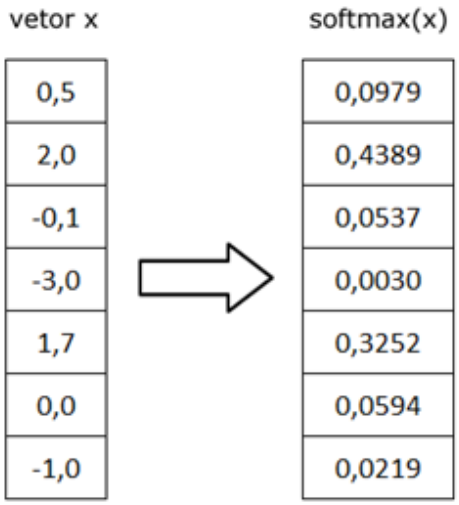


Figura 8 – Exemplo de aplicação da função *softmax* sobre todos os elementos de um vetor

Erro de Entropia Cruzada de Categorias

Como descrito em Goldberg (2016), as funções de erro (*loss functions*), são funções que determinam o erro entre o valor de uma predição e seu valor esperado. O erro $L(\hat{y}, y)$, atribui um valor real para o erro entre a saída gerada $\hat{y}$ e a saída esperada y. No caso da classificação de textos, $\hat{y}$ será a distribuição de probabilidade obtida para cada entrada, enquanto y será, geralmente, um vetor one-hot. Este tipo de vetor é composto por zeros, com exceção de um elemento igual a 1, que indica a classe correta.

O erro de entropia cruzada de categorias também é chamado na literatura de *negative log-likelihood*. Chama-se negativa pois a probabilidade é sempre um valor entre 0 e 1, e então o log será sempre um número negativo ou zero. Dado um exemplo de entrada, este erro mede a dissimilaridade entre a distribuição real (vetor y) e a distribuição da predição (vetor

$\hat{y}$), somando a dissimilaridade em todas as dimensões do vetor para obter o resultado final, conforme a Equação 5.

$$L_{\textit{cross-entropy}}(\hat{y}, y) = -\sum_{i} y_i log(\hat{y}_i) \tag{5}$$

No caso onde o vetor esperado é um vetor *one-hot*, o cálculo do erro de entropia cruzada pode ser simplificado utilizando apenas o valor da inferência correspondente à classe esperada, conforme a Equação 6.

$$L_{\textit{cross-entropy}}(\hat{y}, y) = -log(\hat{y}_i) \tag{6}$$

SGD e Backpropagation

A Descida Estocástica do Gradiente, ou *Stochastic Gradient Descent* (SGD), é um algoritmo geral de otimização, que recebe uma função de erro, os valores de entrada e a saída esperada. Então ajusta os parâmetros da rede, de tal forma que o erro obtido para o exemplo - ou erro médio em um mini-lote - de treinamento diminua.

Para cada exemplo de treinamento: i) efetua-se a predição da distribuição de probabilidades para as classes; ii) calcula-se o erro de acordo com a entropia cruzada das categorias; iii) calcula-se o gradiente do erro em relação aos parâmetros da rede; ou seja, qual é a direção em que cada parâmetro deve ir para que o erro diminua; e iv) ajustam-se os parâmetros usando o gradiente, reduzidos em escala por uma taxa de aprendizado.

O treinamento em mini-lotes é feito calculando o gradiente em cada parâmetro separadamente para cada exemplo, e então a rede é ajustada de acordo com a média dos gradientes para todos os exemplos do mini-lote. Com relação ao tamanho dos mini-lotes, quanto maior for a quantidade de exemplos por lote, melhor será a estimativa do gradiente, pois haverá maior representatividade do conjunto de dados real. E quanto menor for o lote, mais rápida será a convergência do treinamento, pois haverá mais atualizações dos parâmetros (Goldberg, 2016).

O método SGD garante a obtenção de mínimos globais apenas para funções convexas, o que não é o caso das redes neurais profundas. Dessa forma, o SGD aplicado às redes neurais encontra apenas mínimos locais, porém os resultados obtidos são robustos e apresentam boas aplicações práticas, atingindo o estado da arte em diversas áreas.

2.2 Processamento de Linguagem Natural com Redes Neurais

2.2.1 Transfer Learning

Treinar redes neurais profundas é um processo custoso, pois estes modelos grandes precisam processar enormes quantidades de dados para poderem aprender. E além do custo

computacional, muitas vezes são necessárias muitas horas de trabalho humano para a rotulação de dados que serão utilizados durante o aprendizado. O caso torna-se ainda mais oneroso quando a rotulação deve ser feita por especialistas, como a identificação de doenças em imagens médicas ou classificação de textos técnicos. A transferência de aprendizado (*transfer learning*) ajuda a mitigar estes dois problemas.

Para ensinar uma máquina a detectar se o sentimento de uma frase é positivo ou negativo, por exemplo, é necessário fornecer um conjunto de textos associados a um rótulo positivo ou negativo. Uma rede neural é então capaz de aprender representações (*embeddings*) para as palavras do vocabulário, e então relacioná-las para formar uma nova representação da qual seja possível inferir se o texto possui sentimento positivo ou negativo. Com o *transfer learning* é possível aprender as representações de palavras usando um conjunto de textos não-rotulados, e portanto fáceis de se obter, e utilizar o conjunto rotulado apenas para aprender a classificação positiva ou negativa. Dessa forma, torna-se necessário um conjunto menor de textos rotulados, pois a rede já aprendeu muito com os dados não-rotulados, diminuindo assim o custo de se obter dados para o treinamento da rede neural.

Já o custo computacional é reduzido pois modelos pré-treinados podem ser disponibilizados e reutilizados para diferentes tarefas, exigindo um tempo menor de treinamento para a tarefa-fim.

Com as redes neurais recorrentes e *embeddings* estáticos, obtidos por técnicas como o Word2Vec, o padrão era utilizar *embeddings* pré-treinados como inicialização dos vetores em uma determinada RNN. O padrão atual é caracterizado pelo processo de pré-treinamento seguido pelo ajuste (*fine-tuning*) de uma mesma rede. A rede neural é pré-treinada de maneira não-supervisionada com grandes conjuntos de dados não-rotulados, fáceis de obter, e então sofre modificações mínimas em sua arquitetura, e passa pelo processo de *fine-tuning* com dados rotulados, mais custosos de se obter.

No caso do BERT, habilidades como relacionar palavras e formar representações ao nível de sentença são aprendidas em textos livres obtidos da Wikipedia e livros abertos (DEVLIN et al., 2018), e então para o *fine-tuning* introduz-se apenas uma camada de classificação para efetuar a tarefa final do modelo. Dessa forma, ao invés de iniciar o aprendizado para classificação a partir de uma rede neural com pesos totalmente aleatórios, no ponto de partida a rede já sabe relacionar as palavras e formar representações dos textos de entrada, alcançando melhor desempenho na tarefa-fim (GOODFELLOW; BENGIO; COURVILLE, 2016).

No entanto em nem todos os casos o *transfer learning* é feito por meio de treinamento não-supervisionado. Na área de processamento de imagens, o *dataset* ImageNet (DENG et al., 2009), composto por mais de um milhão de imagens distribuídas entre mil classes de objetos e entidades do cotidiano, é comumente utilizado para pré-treinar redes neurais convolucionais que serão depois ajustadas para *datasets* onde é mais custoso de se obter dados rotulados, como raio-x de pulmões indicando a presença de câncer.

2.2.2 Modelagem de Sequências

2.2.2.1 Redes Neurais Recorrentes

As redes neurais recorrentes (RNNs), possuem a característica de processar dados sequenciais (GOODFELLOW; BENGIO; COURVILLE, 2016), tais como os fonemas na fala de uma pessoa, ou as palavras de um texto. Diferentemente do Multi Layer Perceptron, que recebe apenas entradas de tamanho fixo, as RNNs podem lidar com entradas de tamanho variável, o que é possível devido ao compartilhamento de parâmetros: cada elemento da sequência será lido e processado pelo mesmo conjunto de parâmetros, mantendo a cada passo um estado interno que será utilizado no passo posterior.

Uma das concepções mais simples de redes neurais recorrentes é uma rede sem saídas, que apenas processa a entrada e atualiza seu estado oculto, como ilustra a Figura 9, onde x(t) é a entrada no time step "t", que poderia ser cada palavra da frase "o dia está bonito". Ao passo em que a rede processa cada palavra, seu estado interno, representado pela letra "h" (*hidden state*), é atualizado e utilizado na próxima iteração em conjunto com a próxima entrada, conforme a Equação 7, que descreve a recorrência de uma RNN. Desta forma, a rede possui duas matrizes de parâmetros: U para processar a entrada e W para o estado oculto. No time step igual a zero, o estado oculto é inicializado geralmente como um vetor aleatório com valores próximos de zero.

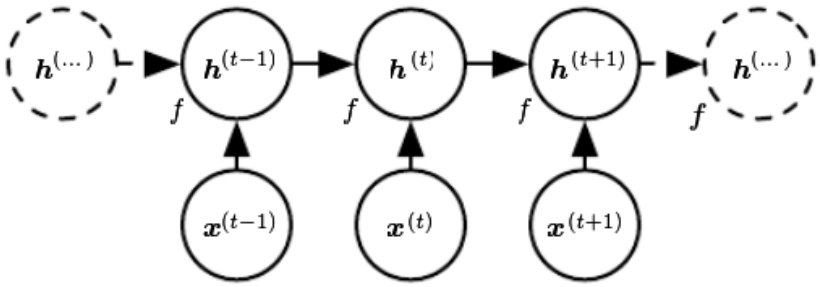


Figura 9 – RNN sem saída - Figura retirada de (GOODFELLOW; BENGIO; COURVILLE, 2016)

$$h^{(t)} = f(h^{(t-1)}, x^{(t)}, \theta) = tanh(b + Wh^{t-1} + Ux^{(t)}) \tag{7}$$

A natureza sequencial das RNNs torna impossível a paralelização da computação ao longo dos *time steps*, pois o estado oculto do *time step* "t" é utilizado na rede no time step "t+1". Durante o treinamento das redes, a retropropagação do erro também deve iniciar do último *time step* e ser sequencialmente calculado até o passo inicial.

A Figura 10 mostra uma RNN para classificação de textos, onde, por exemplo, após ler a frase "o dia está bonito" o modelo deve decidir se esta possui um sentimento positivo ou negativo. Enquanto a rede lê cada palavra da sentença, o estado oculto armazena informações

relevantes para a classificação. Esta rede possui uma matriz de parâmetros adicional, referente à camada de classificação, conforme a Equação 8, cujo resultado da multiplicação com o estado oculto final passa pela ativação *softmax* a fim de transformar a saída em uma distribuição de probabilidades, para então poder calcular o erro segundo a *categorical cross entropy loss*. Esta última operação é representada pela letra L na Figura 10. Cada classe da saída possui um viés associado.

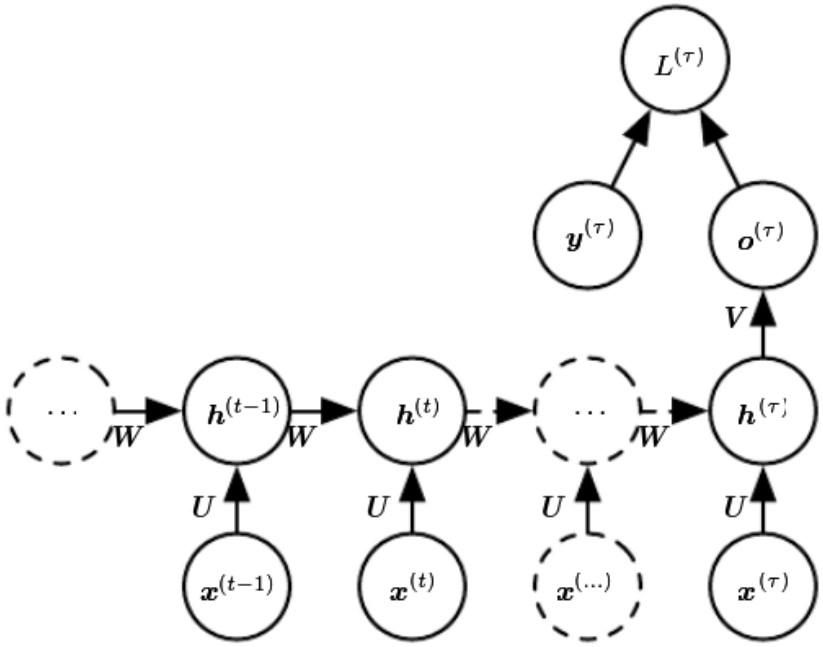


Figura 10 – RNN para classificação - Figura retirada de (GOODFELLOW; BENGIO; COURVILLE, 2016)

$$o^{(t)} = softmax(b + Vh^{(t)}) \tag{8}$$

Além do aspecto computacional, essas recorrências tornam-se um problema para o aprendizado em sequências muito longas, uma vez que a perpetuação de informações por muitos *time steps* é prejudicada pela introdução de novas informações a cada passo. Sequências muito longas também podem sofrer de problemas nos gradientes durante seu treinamento, que comumente podem tornar-se muito pequenos, desaparecendo e tornando-se zero devido à capacidade finita de representação numérica dos sistemas computacionais, ou ainda podem explodir em algumas situações, provocando atualizações dos pesos em escalas prejudiciais ao aprendizado. Para mitigar estes problemas, com exceção do gradiente muito alto, desenvolveu-se a rede *Long Short Term Memory* (LSTM). O problema do gradiente que explode pode ser tratado com uma limitação no valor máximo do gradiente, método chamado de *gradient clipping* (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2.2.2 LSTM

A Figura 11 mostra o diagrama com uma célula LSTM padrão, aplicada a cada *time step*. Na parte inferior da imagem cada nó recebe dois valores, que são o vetor da entrada atual, que pode ser uma nova palavra em uma sentença; e o estado oculto do *time step* anterior. Em cada nó, cada vetor é multiplicado por uma matriz de pesos, somado ao seu viés e transformado pela sigmoide, conforme as Equações 9 a 12. O processo em cada nó é equivalente à soma do resultado de duas camadas totalmente conectadas aplicadas em paralelo.

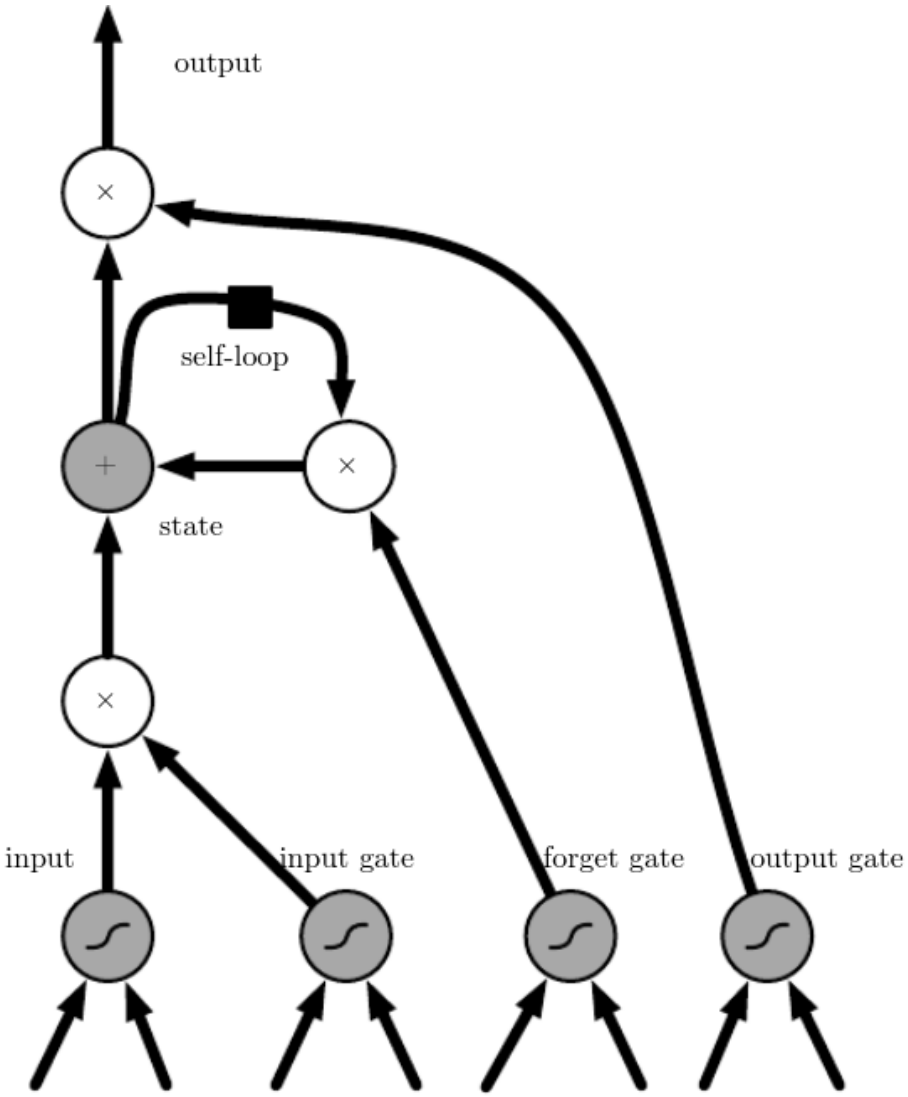


Figura 11 – Célula de uma LSTM - Figura retirada de (GOODFELLOW; BENGIO; COURVILLE, 2016)

$$m^{(t)} = \sigma(b + Ux^{(t)} + Wh^{(t-1)}) \tag{9}$$

$$g^{(t)} = \sigma(b^g + U^g x^{(t)} + W^g h^{(t-1)}) \tag{10}$$

$$f^{(t)} = \sigma(b^f + U^f x^{(t)} + W^f h^{(t-1)}) \tag{11}$$

$$q^{(t)} = \sigma(b^q + U^q x^{(t)} + W^q h^{(t-1)}) \tag{12}$$

A Equação 13 mostra o cálculo da atualização do estado da RNN, onde observa-se o papel das comportas[1] de entrada e de esquecimento. A comporta de entrada decide quanto do vetor de entrada será incorporado ao estado atual da rede, enquanto a comporta de esquecimento define quanto do estado atual será mantido ou esquecido.

$$s^{(t)} = f^{(t)} s^{(t-1)} + g^{(t)} m^{(t)} \tag{13}$$

Já a Equação 14 determina o cálculo do vetor de saída do *time step* "t". A saída é composta pelo estado atual, após aplicação da ativação tangente hiperbólica, multiplicado pelo vetor gerado pela comporta de saída. Calcula-se quanto do estado atual será repassado como saída da rede, dependendo da entrada atual e do contexto armazenado até então.

$$h^{(t)} = tanh(s^{(t)}) q^{(t)} \tag{14}$$

Por meio das comportas de entrada e do esquecimento, o problema de capturar dependências distantes pode ser mitigado pela LSTM, pois alguma característica, codificada na posição "i" do vetor de estado oculto, pode ser mantida caso o portão de entrada apresente um valor próximo de 0 na posição "i" e o portão de esquecimento esteja próximo de 1.

2.2.2.3 Arquitetura Encoder-Decoder

Para tarefas como a classificação de texto, que dá a saída apenas na iteração final, ou a classificação gramatical, que gera uma saída para cada iteração, já sabemos de antemão os tamanhos da entrada e da saída. Porém, em situações como a tradução automática, onde o tamanho da saída não pode ser previsto, é necessária uma adaptação à arquitetura de RNNs apresentada até então. Em 2014, Cho et al. e Sutsveker et al. apresentaram a arquitetura *encoder-decoder*, destinada a problemas que contém uma sentença de entrada e uma sentença de saída, ambas de tamanho variável, também chamada de arquitetura *sequence-to-sequence* (ou seq2seq).

A Figura 12 mostra a arquitetura padrão de uma RNN do tipo *encoder-decoder*. A primeira parte, codificadora, lê cada elemento da sequência, formando ao final um vetor de

[1] Comporta é uma tradução livre de gate, dos termos "input gate", "forget gate" e "output gate". Elas controlam o fluxo de informação, multiplicando os vetores da entrada por números entre 0 e 1, assim como a comporta de uma hidrelétrica, que determina o fluxo de passagem da água entre zero, quando fechada, ou a passagem total da água quando completamente aberta.

contexto "C", que contém informações sobre toda a sequência de entrada. Então a parte decodificadora gera a saída, como a tradução de um texto palavra por palavra. O vetor de contexto é utilizado em cada iteração, juntamente com o estado oculto anterior e o último elemento gerado. Durante todo o processo de geração da saída, o vetor de contexto mantém-se o mesmo.

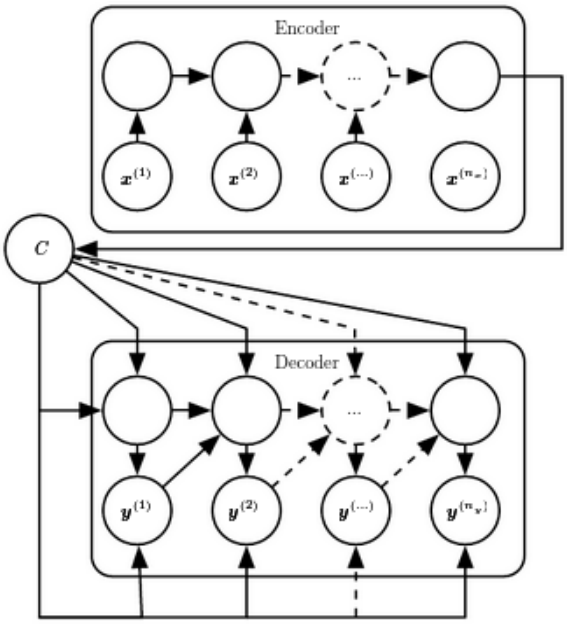


Figura 12 – Arquitetura *encoder-decoder* padrão - Figura retirada de (GOODFELLOW; BENGIO; COURVILLE, 2016)

Diferentemente da tarefa de classificação de textos, onde a rede precisa guardar apenas as informações necessárias para efetuar a classificação, na tradução automática feita pelo *encoder-decoder*, este estado oculto é utilizado como ponto de partida para a geração da tradução, e portanto deve armazenar informações completas sobre toda a sentença de entrada, o que é bem mais difícil (GOODFELLOW; BENGIO; COURVILLE, 2016). Este gargalo de informação levou ao desenvolvimento dos mecanismos de atenção.

Enquanto na arquitetura *encoder-decoder* padrão o vetor de contexto é a única fonte de informação sobre a sequência de entrada, a aplicação da atenção permite a observação direta de dados da entrada a fim de auxiliar o processo de decisão da rede (GOODFELLOW; BENGIO; COURVILLE, 2016). Um dos primeiros mecanismos de atenção foi apresentado por (BAHDANAU; CHO; BENGIO, 2014), aplicado ao problema de tradução automática. No momento de gerar a próxima palavra da sentença traduzida, além de olhar o estado interno, a rede neural recorrente também monta um vetor de contexto próprio para cada time step, composto por meio de uma soma ponderada das representações de todos os *tokens* da entrada. Quem decide o peso dado a cada representação é o mecanismo de atenção, que dá um *score* entre o estado interno atual e cada *token* da entrada, decidindo a importância de cada um naquele *time step*.

A Figura 13 mostra um exemplo apresentado por (BAHDANAU; CHO; BENGIO, 2014), onde são visualizados os pesos dados a cada *token* na língua inglesa no momento de geração de cada *token* na língua francesa, na forma de um mapa de calor. Em geral, cada *token* prestou mais atenção em seu *token* correspondente, mas há dois casos que não seguem este padrão. A entidade "*zone économique européene*" tem seus termos corretamente invertidos na tradução. E o termo "*signed*" usa além do verbo "*signé*", também os termos "*a été*", que apresentam ideia de passado, completando assim o significado do termo em inglês. Esta visualização com mapa de calor é a precursora das visualizações em mecanismos de atenção.

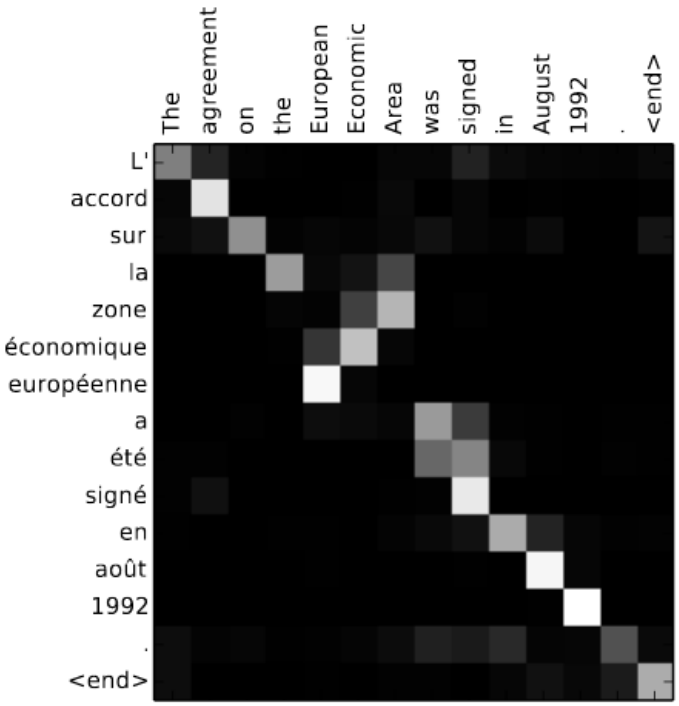


Figura 13 – Visualização da atenção durante a tradução de uma sentença do inglês para o francês - Figura adaptada de (BAHDANAU; CHO; BENGIO, 2014)

2.2.2.4 Transformer

Até o aparecimento do Transformer (VASWANI et al., 2017), os problemas seq2seq eram tratados com redes neurais recorrentes ou convolucionais, incorporadas com mecanismos de atenção. Apresentada em um *paper* intitulado "*Attention is all you need*", esta nova arquitetura dispensa as recorrências, e como sugere o nome da publicação, baseia-se apenas em mecanismos de atenção, mais especificamente a auto-atenção. Este nome vem do fato da atenção ser aplicada sobre as representações internas geradas pelo próprio modelo, e não sobre uma nova observação direta da entrada.

Ao invés de ler as palavras de um texto uma a uma, o Transformer lê todas de uma vez. Dessa forma soluciona-se o problema de capturar relações entre itens distantes uns dos outros, assim como o problema da paralelização, pois cada item da sequência está diretamente

conectado a todos os demais itens, como mostrado na Tabela 1. O caminho máximo entre dois itens da sequência, igual ao tamanho da entrada para as RNNs, é constante e igual a um para a camada de auto-atenção, assim como a quantidade de operações em sequência. A Tabela 1 ainda mostra a complexidade, ou seja, o custo computacional, de cada camada. A complexidade do Transformer só é maior do que as redes recorrentes quando n>d, onde n é o tamanho da entrada e d a quantidade de dimensões da representação do modelo. Mas a grande vantagem é o alto poder de paralelização do Transformer, que permite treinar com mais exemplos no mesmo tempo.

Tabela 1 – Comparação entre a camada recorrente das RNNs e a camada de auto-atenção do Transformer - Tabela adaptada de (VASWANI et al., 2017)

Tipo de Camada	Complexidade por Camada	Operações Sequenciais	Tamanho do Caminho Máximo
Recorrente (RNN)	$O(n.d^2)$	$O(n)$	$O(n)$
Auto-Atenção (Transformer)	$O(n^2.d)$	$O(1)$	$O(1)$

A Figura 14 ilustra a arquitetura Transformer, com a camada *encoder* à esquerda e *decoder* à direita. A codificação é feita em duas etapas: primeiro aplica-se a operação de auto-atenção múltiplas vezes - uma vez para cada cabeça de atenção; e em seguida as representações são transformadas como em uma camada totalmente conectada. Entre cada etapa são inseridas conexões residuais, e após cada etapa os vetores são somados com os adventos das conexões residuais e normalizados. A entrada da camada de codificação corresponde aos *embeddings* dos *tokens* da sentença de entrada somados a um *embedding* posicional, cuja finalidade é adicionar codificações temporais nos *embeddings*, permitindo que o modelo saiba a ordem dos *tokens*. A entrada das camadas possui dimensões n x d.

Operação de Auto-Atenção

Em cada cabeça de atenção, a entrada (de dimensão n x d) é multiplicada por três diferentes matrizes de pesos (W^Q, W^K, e W^V), gerando três matrizes: Q, K e V, cada uma com dimensões n x k, onde k é o tamanho da representação gerada por cada cabeça. As matrizes de peso W^Q, W^K, e W^V projetam a entrada em um subespaço de representação. Q refere-se ao termo *query*, K refere-se a *key*, e V é referente a *value*. No *paper* original do Transformers (VASWANI et al., 2017) utiliza-se $d = 512$ e $k = 64$.

A Figura 15, retirada do ótimo artigo[2] explicativo "*The Illustrated Transformer*" mostra o processo de codificação para o *token* "*Thinking*" da sentença "*Thinking Machines*", com apenas duas palavras para simplificar o exemplo. Neste caso, n=2, d=4 e k=3. A partir dos *tokens* da entrada já codificados, obtém-se os vetores Q, K e V de cada palavra. Para calcular a atenção que o *token* "*Thinking*" deve prestar em cada *token* da entrada, toma-se o produto vetorial entre o vetor *query* do *token* "*Thinking*" e o vetor *key* de cada *token* da entrada -

[2] http://jalammar.github.io/illustrated-transformer/

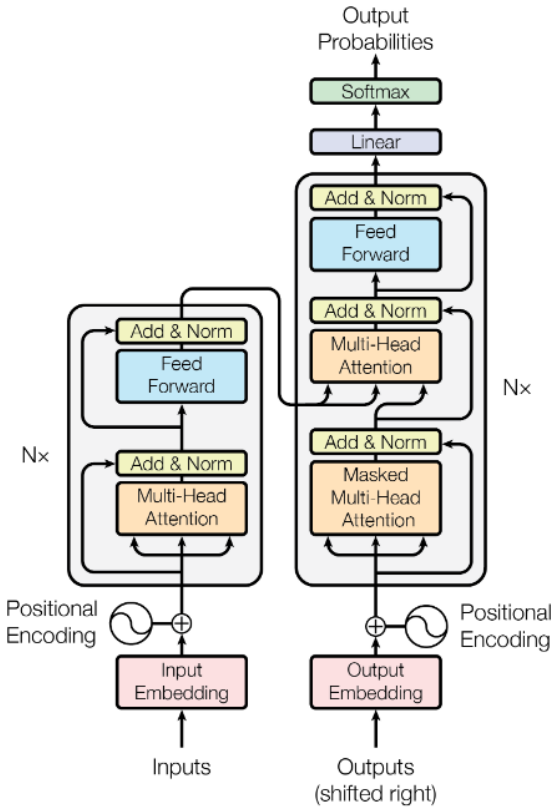


Figura 14 – Arquitetura do Transformer. *Encoder* à esquerda e *decoder* à direita. Figura retirada de (VASWANI et al., 2017)

apenas dois neste caso, resultando nos scores. Quanto maior a similaridade entre o *token query* de "*Thinking*" e o *token key* dos demais *tokens*, maior será o *score*. Vaswani et al. alegam que é importante reduzir a escala do *score* a fim de estabilizar os valores da função *softmax*, para isso eles o dividem pela raiz quadrado do tamanho da representação de cada cabeça de atenção (64 no caso do *paper*), e portanto a Figura 15 mostra a divisão por oito após o cálculo do *score*.

O processo segue com a aplicação da função *softmax* sobre o vetor de scores do *token* "*Thinking*". A cabeça de atenção utilizada, quando construindo a nova representação do *token* "*Thinking*", decidiu prestar muito mais atenção no próprio *token* do que em "*Machines*", o que está ilustrado na linha "*Softmax* x *Value*". Ao final do processo, a nova representação do *token* "*Thinking*" é 88% do vetor *value* de "*Thinking*" somada a 12% do vetor *value* do *token* "*Machines*". Para obter o novo *token* de "*Machines*" aplica-se o mesmo processo, mas agora os produtos escalares envolvem o vetor *query* de "*Machines*" com os vetores *key* dos demais *tokens*.

Para utilizar múltiplas cabeças de atenção, simplesmente concatena-se o resultado de cada uma e multiplica-se por uma quarta matriz de pesos. Esta saída é somada aos *embeddings* de entrada, por meio da conexão residual, e normalizados, para então seguirem para uma camada totalmente conectada comum, e após nova soma residual e normalização, partirem

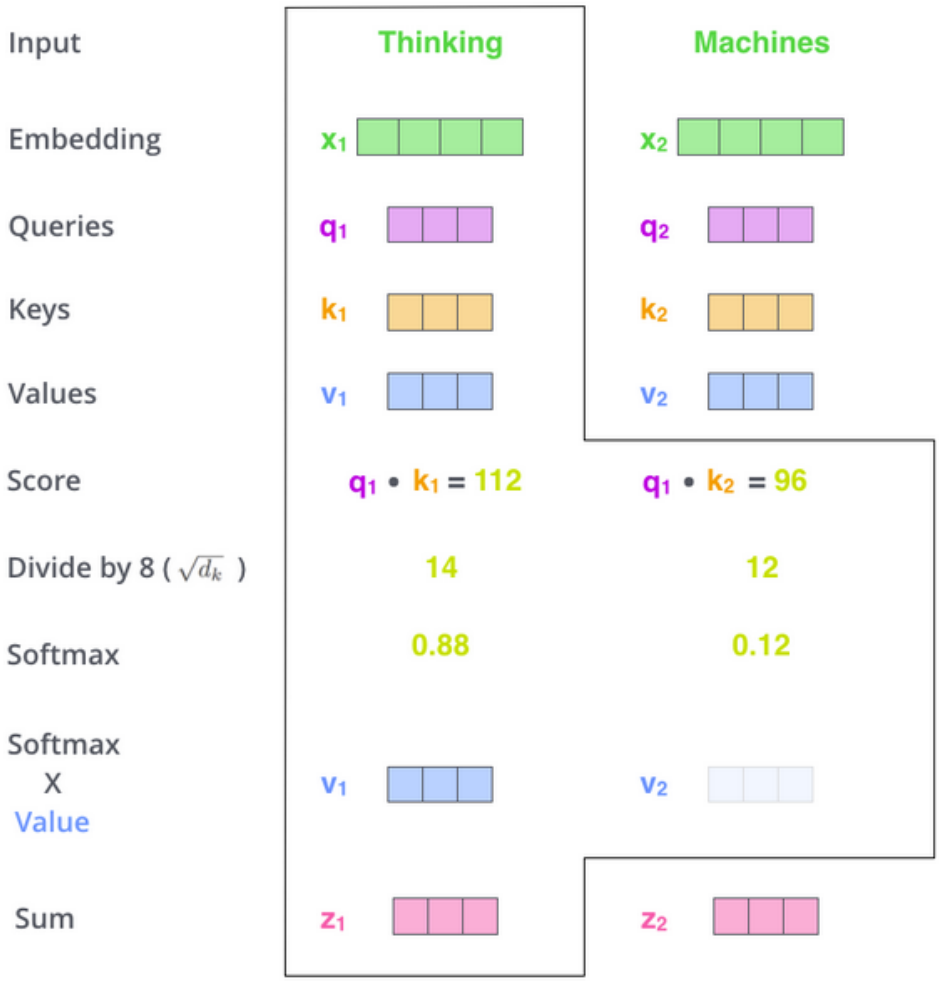


Figura 15 – Cálculo da auto-atenção para o *token* "*Thinking*". Figura retirada do artigo "The Illustrated Transformer"

para a próxima camada de codificação. No *paper* original são empilhados seis *encoders*, cuja saída é utilizada pelos *decoders* da maneira ilustrada na Figura 16.

Decoder

A camada de decodificação tem como entrada a sentença gerada até então. No caso da Figura 16, poderia ser "*I am a*", enquanto "*student*" seria a palavra esperada. Esta camada é composta por três subcamadas. Um mecanismo de auto-atenção mascarado, similar à auto-atenção explicada anteriormente, mas com a diferença de desconsiderar as palavras não vistas da entrada. A palavra "*student*" é tratada com score $-\inf$, e assim o *softmax* resulta em 0 nesta posição, e portanto ela nunca é usada. Outra subcamada é uma atenção *encoder-decoder*, onde as matrizes K e V vêm do último codificador, e Q vem da saída gerada até então. E por fim uma camada totalmente conectada. Como o primeiro mecanismo de atenção possui a matriz V vinda da saída gerada até então, e o segundo mecanismo possui a matriz V vinda do *encoder*, e portanto da entrada, informações de ambos são utilizados para formar a saída da camada de decodificação.

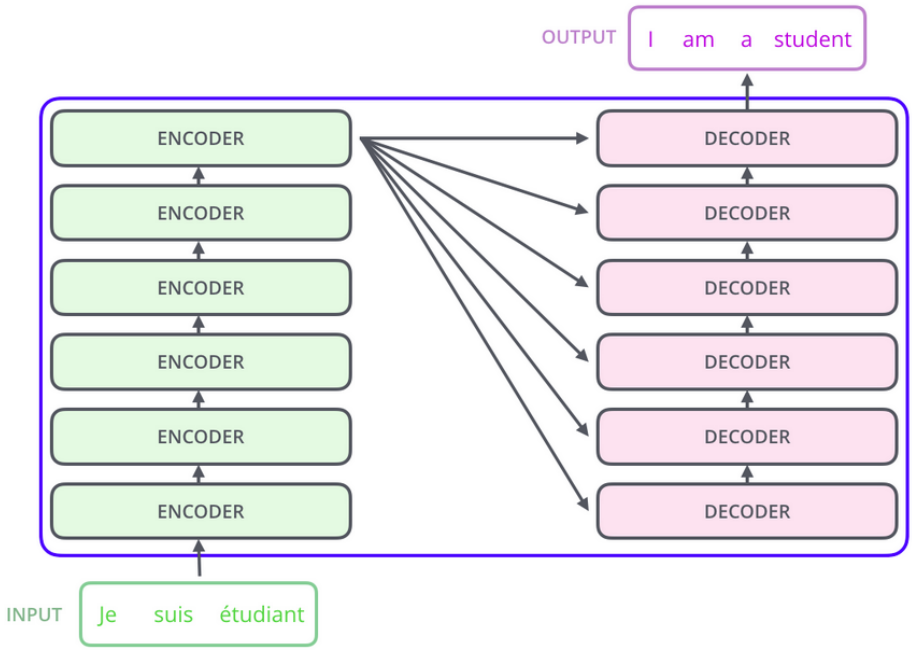


Figura 16 – Uso da saída dos codificadores nos decodificadores - Imagem retirada do artigo "The Illustrated Transformer"

Embeddings Posicionais

Para permitir que o modelo saiba a posição de cada *token* na sentença, ou ainda a distância entre dois *tokens*, soma-se aos *embeddings* de entrada *embeddings* que indicam a posição. São utilizados vetores fixos, de acordo com as senoides das Equações 15 e 16, onde "pos" é a posição do *token* na sentença (de 0 a n-1) e i é a dimensão. Estas funções permitem que, para qualquer deslocamento k, PE (pos + k) possa ser escrito como uma função linear de PE(pos), e assim o modelo consegue captar a distância entre os *tokens*.

$$PE_{pos,2i} = sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{15}$$

$$PE_{pos,2i+1} = cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}}) \tag{16}$$

Vaswani et al. (2017) também experimentaram o uso de *embeddings* posicionais que são aprendidos durante o treinamento, em oposição às senoides fixas, onde obtiveram igual performance. Portanto escolheram utilizar vetores fixos, uma vez que estes podem ser facilmente estendidos para entradas maiores do que as vistas durante o treinamento, o que seria impossível para *embeddings* aprendidos para cada posição.

2.2.2.5 BERT

O BERT (*Bidirectional Encoder Representations from Transformers*), apresentado por (DEVLIN et al., 2018), é uma rede neural para processamento de linguagem natural,

composta por 12 camadas *encoder* do Transformer empilhadas, dispensando totalmente a parte decodificadora, mais utilizada por modelos geradores de textos, como o GPT-2 (RADFORD et al., 2019). O BERT aproveita-se do Transfer Learning ao pré-treinar seus parâmetros em grandes conjuntos de textos de maneira não-supervisionada, exigindo pequenas modificações para executar tarefas em um domínio específico, como a inserção de uma camada de classificação seguida por um treinamento em dados rotulados.

Ao invés de pré-processar os textos ao nível de caracteres ou palavras, a entrada do BERT é dividida em sub-unidades de palavras, conforme a abordagem Wordpiece (WU et al., 2016), que lida bem com palavras fora do vocabulário. Com esta abordagem, a sentença "o dia está bonito" pode ser convertida para "o dia est ##á bo ##nito", por exemplo. Dessa forma, a sentença que tem quatro palavras torna-se uma entrada para o modelo formada por seis sub-unidades de palavras. Além disso, adicionam-se dois *tokens* especiais: o "[CLS]" e o "[SEP]".

O *token* "[CLS]" é adicionado no início de todas as entradas, sendo utilizado como uma representação agregada de todo o texto para classificações ao nível de sentença. Já o *token* "[SEP]" é utilizado para separar diferentes sentenças em uma mesma entrada, provendo versatilidade ao BERT. Com este *token*, o modelo pode efetuar tarefas que consideram apenas uma sentença, como a análise de sentimento de um texto, assim como tarefas que envolvem pares de textos, como encontrar a resposta à uma pergunta, onde a pergunta e o trecho onde encontra-se a resposta são separados pelos *token* "[SEP]", que também é repetido ao final da entrada. A Figura 17 ilustra um exemplo de texto pré-processado.

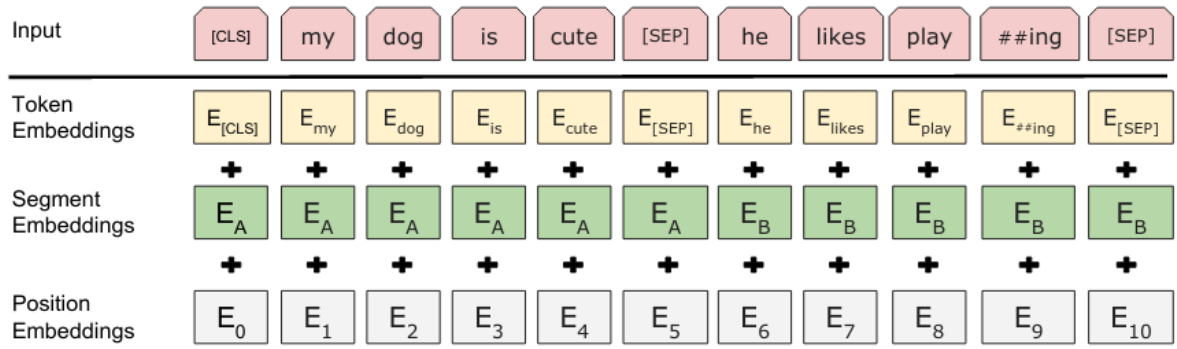


Figura 17 – Pré-processamento textual do BERT e formação dos *embeddings* na primeira camada. Figura retirada de (DEVLIN et al., 2018)

A Figura 17 ainda mostra o processo de formação dos *embeddings* na primeira camada do BERT. Cada sub-unidade possui um *embedding* próprio, chamado de "*Token Embedding*" na Figura 17. Para identificar à qual parte da entrada cada *token* pertence, o modelo aprende um *embedding* para o segmento textual A, que vai até o primeiro *token* "[SEP]", e outro *embedding* para o segmento B. O BERT também utiliza o *embedding* posicional apresentado na Seção anterior para identificar a posição relativa entre os *tokens*. Como entrada do primeiro

codificador, a representação de cada *token* é composta pela soma dos três diferentes tipos de *embedding*.

O pré-treinamento do BERT é baseado em duas tarefas: i) Modelo de Linguagem Mascarado (MLM); e ii) Predição de Próxima Sentença (NSP, da sigla em inglês para *Next Sentence Prediction*). A Figura 18 mostra dois exemplos destas tarefas. De maneira simplificada, a primeira tarefa consiste em pegar uma sentença tokenizada e substituir 15% dos *tokens* por um *token* especial, o "[MASK]", e então o modelo deve aprender a preencher a sentença corretamente. Enquanto modelos de linguagem baseados em RNNs só conseguem utilizar os *tokens* anteriores para fazer uma predição sobre o *token* atual, a arquitetura Transformer permite que a rede olhe para todos os *tokens* da sentença ao mesmo tempo. Daí vem o nome "Bidirecional": as RNNs só enxergam em uma direção do contexto, enquanto os Transformers olham o contexto à esquerda e à direita[3].

A tarefa de NSP consiste em, dadas duas sentenças, dizer se a segunda é uma sequência natural da primeira. Devlin et al. (2018) alegam que ela é importante para tarefas-fim ao nível de sentença, como *question-answering*, onde deve-se extrair uma resposta de um texto; e inferência de linguagem natural, onde, dada uma premissa e uma hipótese, verifica-se se a hipótese é contraditória à premissa, se é uma implicação da premissa, ou se é neutra, ou seja, sobre um assunto não relacionado. No entanto demonstrou-se que a tarefa de NSP pode ser muitas vezes prejudicial ao desempenho (AROCA-OUELLETTE; RUDZICZ, 2020), levando à proposta de variações do BERT que não se utilizam desta tarefa, e que também utilizam variações do MLM apresentado anteriormente, como RoBERTA (LIU et al., 2019) e XLNET (YANG et al., 2019).

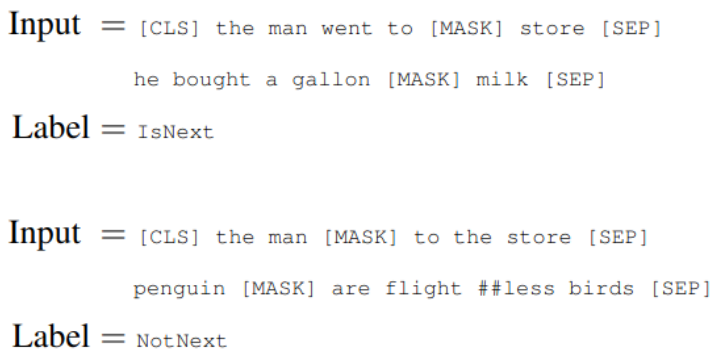


Figura 18 – Exemplo das duas tarefas de pré-treinamento do BERT. Figura retirada de (DEVLIN et al., 2018)

A Google disponibiliza modelos pré-treinados em diversas configurações, como modelos próprios para inglês e chinês, assim como multilinguais[4], treinados em quantidades enormes de textos. O pré-treinamento em inglês, por exemplo, é feito sobre um conjunto de livros, o

[3] Ainda que existam RNNs bidirecionais, estas são formadas pela junção de uma recorrência que percorre a entrada da esquerda para a direita, e outra no sentido contrário, mas não nos dois sentidos ao mesmo tempo.

[4] Os modelos pré-treinados estão disponíveis em https://github.com/google-research/bert

BookCorpus, e a Wikipedia inteira, totalizando três bilhões e 300 milhões de palavras, levando quatro dias para ser completado em uma configuração potente de GPUs. Dessa forma, o habitual é utilizar o modelo pré-treinado pela Google e fazer apenas o ajuste da rede utilizando dados próprios. No entanto em domínios específicos o pré-treinamento com textos próprios mostra-se mais vantajoso, o que levou ao surgimento de outros BERTs, como o BioBERT (LEE et al., 2020).

2.2.3 Interpretabilidade em Processamento de Linguagem Natural

2.2.3.1 XAI (eXplainable Artificial Intelligence)

Estamos acostumados com inteligências artificiais recomendando filmes, produtos, e até mesmo amigos em redes sociais. Os modelos recentes vêm apresentando resultados excelentes, porém são cada vez menos interpretáveis devido à sua complexidade. Em decisões críticas, como um diagnóstico médico, é imprescindível que saibamos as razões que levaram a IA a realizar um determinado diagnóstico. Daí a importância de XAI: garantir confiança nos sistemas de IA de alta performance, para que assim possamos incorporá-los em nossas atividades de maneira segura (Adadi; Berrada, 2018).

Adadi e Berrada (2018) destacam quatro razões que remetem à importância de XAI: i) **justificar** as decisões tomadas, a fim de ganhar a confiança dos usuários; ii) **controlar** o processo de decisão do modelo, permitindo assim a correção de falhas; iii) **melhorar** o desempenho do modelo, o que torna-se possível ao saber o porquê de suas decisões; e iv) **descobrir** novos conhecimentos que a IA pode ter adquirido durante o treinamento, e que ainda não seja um conhecimento do especialista que a utiliza.

Historicamente os métodos de processamento de linguagem natural sempre foram inerentemente explicáveis, pois tratavam-se da aplicação de processos desenvolvidos por especialistas, como a aplicação de regras gramaticais e árvores de decisão (DANILEVSKY et al., 2020). Estas técnicas são também chamadas de caixas-brancas (do termo *white-box* em inglês), pois entre a entrada e a saída dos modelos, todo o processo pode ser conhecido e compreendido a partir de sua formulação.

Atualmente, modelos de redes neurais profundas, baseados na arquitetura Transformer, compõem o estado da arte em diversas tarefas de processamento de linguagem. Porém, neste caso, as razões que levam a uma determinada saída são desconhecidas. Ou seja, entre a entrada e a saída do modelo, o processo de tomada de decisão não é totalmente explicado pelos especialistas. Por este motivo, estas técnicas são conhecidas como caixas-pretas (*black-box*).

2.2.3.2 Categorias para Explicação

Danilevsky et al. (2020) apresentam diferentes categorizações para os métodos de explicação de modelos de inteligência artificial. Duas grandes categorias, que eles trazem de

trabalhos anteriores, são o escopo da explicação (global ou local), e se o método precisa de algum processamento adicional à inferência (*self-explanation* ou *post-hoc*).

Uma explicação local busca interpretar o modelo a partir da predição de uma instância específica. Ou seja, o modelo é explicado por meio do que acontece com uma determinada entrada. Já as explicações globais buscam explicar o funcionamento do modelo sem focar em nenhuma entrada em particular.

Self-explanation refere-se a explicações que utilizam dados gerados durante o processo de inferência, enquanto as técnicas *post-hoc* exigem um processo adicional. Os valores produzidos pelos mecanismos de atenção são do tipo *self-explanation*, pois tratam-se da visualização de informações que são obtidas durante a predição. Já uma análise da importância de cada característica de um espaço de representações, como a aplicação de informação mútua, trata-se de um processo *post-hoc*.

2.2.3.3 Categorias para Explicação em NLP

As duas categorizações discutidas na Seção anterior são bem abrangentes em relação à explicabilidade. Danilevsky et al. (2020) percebem que a área de interpretabilidade em processamento de linguagem natural carece de categorizações específicas, e então fornecem outras três categorias para separar os trabalhos da área: i) referente à técnica de explicabilidade; ii) referente à operação que possibilita a explicação; e iii) referente à visualização utilizada. Cada categoria é brevemente apresentada na sequência, com foco no problema de justificar a predição de um modelo.

Técnicas de Explicabilidade

As cinco técnicas principais utilizadas são: importância das características; modelos substitutos; exemplos similares; procedência; e indução declarativa.

As técnicas de importância das características podem investigar tanto as partes mais importantes da entrada, como as palavras mais relevantes um em texto ou pixels em uma imagem, ou ainda características de representações internas de redes neurais. Já os modelos substitutos são modelos interpretáveis (geralmente lineares), que são treinados para imitar a saída do modelo mais complexo em análise. Uma vez que o funcionamento dos modelos substitutos é muito diferente do modelo estudado, a efetividade desta última técnica é bem questionada na literatura (DANILEVSKY et al., 2020).

A exibição de exemplos similares consiste em apresentar os exemplos semanticamente mais próximos da instância investigada, sendo comumente utilizado na classificação de textos.

As técnicas baseadas em procedência - *provenance-based* - visam a explicação de todo o passo a passo que leva a uma predição. São muito utilizadas para a tarefa de *question-answering*. E a indução declarativa ocorre por meio da geração de explicações como regras ou árvores de decisão.

O presente projeto investigará técnicas relacionadas à importância das características, assim como o uso de exemplos similares.

Operações para Explicabilidade

Seis operações são destacas por Danilevsky et al. (2020): saliência da primeira derivada; propagação da relevância ao nível das camadas; perturbação da entrada; atenção (já discutida na Seção 2.2.2.2); sinais das comportas em LSTMs; e *design* de arquiteturas voltadas à explicabilidade.

A saliência da primeira derivada consiste em obter a derivada parcial de cada parte da entrada em relação à saída, estimando assim a contribuição de cada parte da entrada na predição. Esta técnica é amplamente utilizada em redes neurais pois as bibliotecas já fornecem os meios para este cálculo, uma vez que faz parte do processo de treinamento dos modelos.

A propagação da relevância ao nível das camadas busca identificar a importância de características em camadas intermediárias das redes neurais. Já a perturbação de entrada visa à investigação do efeito de alterações na entrada, observando o que acontece com a saída dos modelos.

Os sinais das comportas em LSTMs são complementares às saídas das LSTMs, permitindo assim a visualização de informações adicionais que podem influenciar o funcionamento deste modelos recorrentes. E o design de arquiteturas voltadas à explicabilidade consiste no desenvolvimento de arquiteturas que seguem o raciocínio humano, e por estarem mais próximas do nosso pensamento, são mais facilmente interpretáveis.

Visualizações

Danilevsky et al. (2020) destacam três formas de exibição de explicações para os usuários: saliência; representações declarativas; e explicações em linguagem natural.

A visualização da saliência - importância - é geralmente dada na forma de mapas de calor, ou ainda na forma de cores em diferentes intensidades. Estão diretamente relacionadas à técnica de importância de características. Já as representações declarativas são a exibição de árvores de decisão ou regras, possibilitadas pelas operações de indução declarativa. E as explicações em linguagem natural envolvem modelos geradores de texto, que buscam explicar as predições em texto livre, e portanto próximas a forma como um humano explicaria.

2.2.3.4 Dificuldades

Uma das grandes deficiências na área de interpretabilidade para modelos modernos de processamento de linguagem natural é a falta de padrões de avaliação das técnicas (DANILEVSKY et al., 2020). Em grande parte isso ocorre por tratar-se de uma área desenvolvida recentemente, mas também conta o fato de potencialmente haver mais de uma explicação em cada caso analisado, além de os modelos aplicarem uma extensa sequência de operações até

chegarem ao resultado final.

Danilevsky et al. (2020) apontam que dentre os 50 trabalhos analisados em sua *survey*, 32 apresentaram apenas avaliações informais das técnicas, ou nem mesmo avaliação alguma, enquanto apenas nove realizaram avaliações com humanos. Outro problema é que a maioria dos trabalhos não são claros quanto a o quê está sendo avaliado, e se as explicações são fiéis ao processo de inferência dos modelos ou facilmente compreendidas por humanos.

A cobertura das técnicas também é um problema. Muitas explicações, como a redução da entrada, olham apenas a entrada e saída dos modelos, deixando todo o funcionamento interno para a imaginação do usuário (DANILEVSKY et al., 2020). Outras explicações visam a interpretação de apenas uma parte do processo, enquanto o ideal seria prover explicações de todo o passo a passo, da entrada à predição.

2.2.3.5 Interdisciplinaridade

As técnicas de XAI geralmente são desenvolvidas por especialistas em computação, os quais levam em consideração apenas sua intuição sobre o que torna uma técnica de interpretação boa. Miller, Howe e Sonenberg (2017) destacam que há um amplo conhecimento em estudos comportamentais humanos, pautados em psicologia, sociologia e ciências sociais, que poderia ser aproveitado pelos desenvolvedores de métodos de XAI a fim de criar melhores explicações e avaliações. Em particular, deveriam ser considerados estudos sobre como os seres humanos explicam assuntos para outros seres humanos.

Perguntas sobre o porquê de algo ter acontecido, como porque um classificador escolheu a classe "A", são implicitamente contrastivas. Ou seja, ao perguntar a razão do classificador ter escolhido a classe "A", também está se perguntando o motivo de não ter-se escolhido a classe "B"(MILLER; HOWE; SONENBERG, 2017). Dessa forma, para explicar a predição de um modelo, é interessante também mostrar uma explicação contrastiva, que mostre o que teria levado a uma saída diferente. Este tipo de explicação também é mais simples de ser feita, pois o usuário do modelo pode se dar satisfeito ao entender a diferença entre os dois casos, sendo desnecessária uma explicação completa de todos os eventos que levaram à classificação "A".

Foco nos pontos principais: uma explicação resumida, pontuando uma ou duas causas principais, é melhor recebida do que explicações completas e cheias de detalhes. No entanto, Miller, Howe e Sonenberg (2017) também destacam que uma boa explicação é similar a uma conversa, portanto a interação é essencial para permitir que o usuário aprofunde sua análise de acordo com o seu conhecimento e interesse na explicação. Assim, uma boa técnica de XAI deve começar com os pontos principais da explicação, mas permitir que o usuário interaja com o sistema em busca de maiores explicações.

3 Trabalhos Relacionados

O início desta Seção está dividida entre técnicas de entendimento que independem do modelo, porém aplicadas ao BERT, e técnicas desenvolvidas especificamente para a interpretação do Transformer ou do BERT. Ao final são apresentados trabalhos cuja finalidade é garantir a qualidade das anotações em datasets rotulados.

3.1 Técnicas Agnósticas ao Modelo

O AllenNLP Interpret[5] (WALLACE et al., 2019) é um *framework* que oferece duas técnicas que independem do modelo, ou seja, não são específicas para o BERT. Uma técnica, baseada em ataques adversariais, efetua perturbações na entrada a fim de testar os limites de predição, e a outra, baseada em saliência (SIMONYAN; VEDALDI; ZISSERMAN, 2014), determina a importância de cada *token* do texto de entrada para a classificação. Permite-se a análise de modelos para diversas tarefas de classificação, como o Modelo de Linguagem Mascarado apresentado na Seção 2.2.2.5 e a classificação de textos.

3.1.1 Perturbação da Entrada

As técnicas de perturbação da entrada visam à observação da predição do modelo em diferentes variações de uma mesma entrada. O AllenNLP Interpret permite dois tipos de ataques adversariais baseados em perturbações na entrada: a redução da entrada e o Hot-Flip (EBRAHIMI et al., 2018).

A redução da entrada é feita removendo *tokens* contíguos da sentença original, até encontrar a menor sentença que continua com a mesma predição da sentença original. Dessa forma, encontra-se o trecho do texto ao qual pode-se atribuir a classificação final, motivo pelo qual tais técnicas também são chamadas de métodos de atribuição. A Figura 19 traz um exemplo retirado do artigo original de Wallace et al. (2019). Na sentença completa, "NLP *Cool*" foi classificado como uma pessoa (PER). A menor entrada para a qual "NLP *Cool*" continua sendo pessoa é "*named* NLP *Cool*". Por meio desta análise foi possível verificar que o termo "*named*" antecedendo "NLP *Cool*" foi suficiente para determinar que trata-se de uma pessoa.

A técnica Hot-Flip consiste em substituir *tokens* da entrada e checar a classificação obtida, ou ainda substituir *tokens* visando a mudança na predição para uma classe específica. Desta última forma, ela pode dar suporte a explicações contrastivas, potencialmente fornecendo explicações mais convincentes.

[5] https://allennlp.org/interpret

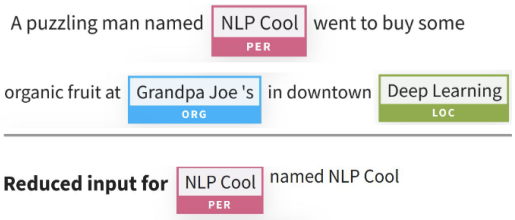


Figura 19 – Texto atribuído à classificação de "NLP Cool" como uma pessoa. Figura adaptada de (WALLACE et al., 2019)

Ao invés de cegamente substituir cada token de forma aleatória, a técnica Hot-Flip primeiro encontra a palavra mais importante para a predição atual, e então foca sua substituição nesse token, com exceção de stopwords, que quando substituídas podem facilmente modificar a semântica do texto (EBRAHIMI et al., 2018). A partir daí, para garantir que a semântica continue a mesma após a substituição, o novo token deve ter similaridade cossenoidal maior que 0,8 em relação ao token a ser substituído, e ambos também devem ser da mesma classe gramatical. Um processo de beam search é efetuado para determinar múltiplas trocas de tokens sobre uma mesma entrada.

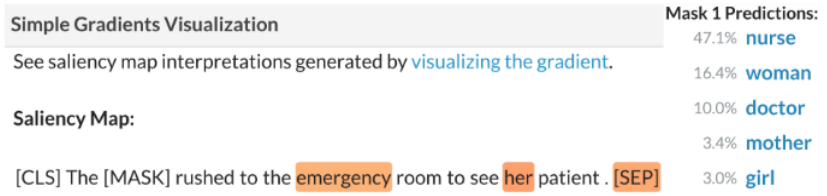
Abnar e Zuidema (2020) também efetuam um ataque adversarial para determinar a importância de cada token da entrada, chamado de blank-out. A técnica consiste em substituir cada token da entrada pelo token "UNK" (*unknown*), e então observar a alteração no score obtido para a classe correta, comparado ao obtido na sentença original. Quanto maior a alteração no score, mais importante o token é considerado.

3.1.2 Mapa de Saliência

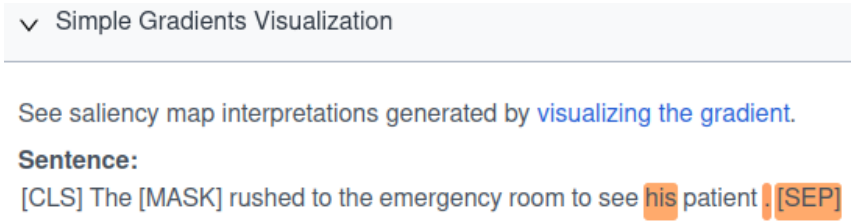
Os mapas de saliência (SIMONYAN; VEDALDI; ZISSERMAN, 2014) também buscam mostrar quais partes da entrada são mais relevantes para a predição de uma determinada saída, porém ao invés de realizar ataques adversariais, observam o gradiente em cada parte da entrada em relação à classe de interesse. Originalmente os mapas de saliência foram propostos para imagens, onde cada parte da entrada corresponde a um pixel, porém a técnica pode ser aplicada para textos ao considerar cada *token* como uma parte da entrada. A diferença é que ao invés de obter um mapa de saliência, portanto com duas dimensões, o método resulta em um vetor de saliência, unidimensional e do tamanho da entrada.

O trabalho inicial sobre mapas de saliência (SIMONYAN; VEDALDI; ZISSERMAN, 2014) aproxima a função da rede neural (S_c) como uma expansão de primeira ordem de uma série de Taylor, conforme a Equação 17, onde w é a derivada do *score* para a classe de interesse em relação a cada pixel de uma imagem de entrada "I".

$$S_c(I) \approx w^t I + b \tag{17}$$



(a) retirada de (WALLACE et al., 2019)



(b) obtida por meio da ferramenta disponibilizada online[a]

[a] https://demo.allennlp.org/masked-lm

Figura 20 – Viés de gênero para profissões detectado pelo mapa de saliência no AllenNLP Interpret.

Para obter a derivada w, basta efetuar o processamento normalmente a partir da entrada, e então fazer a retropropagação (backpropagation). Mas ao invés de fazer a retropropagação em relação ao erro de classificação, como é feito durante o treinamento, a saliência é obtida a partir do logit de saída na posição correspondente à classe de interesse. Dessa forma obtém-se quais os tokens que, quando alterados, mais refletem na alteração do score para a classe desejada.

Além do método padrão, também chamado de Vanilla Gradient na literatura, o AllenNLP Interpret ainda disponibiliza outras duas formas de obtenção dos mapas de saliência: os Gradientes Integrados (SUNDARARAJAN; TALY; YAN, 2017) e o SmoothGrad (SMILKOV et al., 2017).

Na Figura 20a, retirada de Wallace et al. (2019), apresenta-se o mapa de saliência para a predição do token mascarado, onde os tokens "emergency", "her", e o token especial "[SEP]" foram considerados os mais relevantes para a predição do token "nurse". A Figura 20b apresenta as predições para uma sentença bem similar, porém substituindo "her" por "his". Fica evidente que o modelo considera que a contração "dela" tem muito mais a ver com a profissão de enfermeiro do que de médico, indicando um viés de gênero no modelo, que poderia ser combatido com uma reformulação do conjunto de treinamento, colocando mais casos com enfermeiros homens e médicas mulheres.

Os Gradientes Integrados (SUNDARARAJAN; TALY; YAN, 2017) ponderam a importância de cada parte da entrada utilizando uma entrada baseline, considerada como sem informação, que para imagens poderia ser uma imagem toda preta, e para textos poderia ser um vetor apenas com zeros. Acumulam-se os gradientes considerando várias entradas resultantes de uma mistura entre a entrada investigada (x) e o baseline (x'), conforme a Equação 18,

onde F é a função da rede neural.

Computacionalmente, o cálculo é realizado conforme a Equação 19, evidenciando que trata-se de um acúmulo dos gradientes considerando m diferentes formas da entrada, ponderando a importância de cada *feature*. m é um parâmetro escolhido pelo usuário, que determina a quantidade de representações intermediárias utilizadas entre o baseline e a entrada original.

$$IntegratedGrads_i(x) = (x_i - x'_i) * \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x - x'))}{\partial x_i} * d\alpha \tag{18}$$

$$IntegratedGrads_i^{approx}(x) = (x_i - x'_i) * \sum_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m}(x - x'))}{\partial x_i} * \frac{1}{m} \tag{19}$$

Segue na Figura 21 uma aplicação dos Gradientes Integrados para identificar as partes mais importantes de um texto em um problema de classificação de questões, onde o modelo precisa classificar corretamente qual é o tipo de resposta esperado pela pergunta.

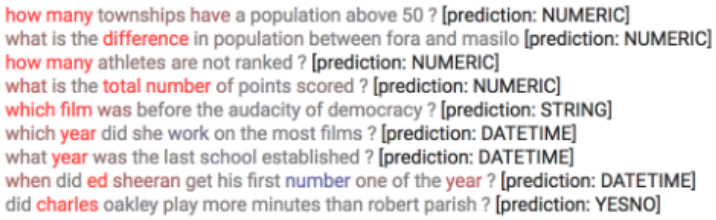


Figura 21 – Detecção das porções mais relevantes para a classificação, obtidas pelo método dos Gradientes Integrados. Retirado de (SUNDARARAJAN; TALY; YAN, 2017)

A técnica SmoothGrad (SMILKOV et al., 2017) também utiliza diversas vezes a entrada modificada, porém obtém diferentes versões dela ao aplicar a adição de ruído, podendo ser utilizada em conjunto com qualquer método de cálculo do mapa de saliência. Obtém-se um mapa de saliência para cada versão da entrada após a adição de ruído, e então toma-se a média dos mapas como o resultado final. O AllenNLP Interpret implementa o SmoothGrad como adição de ruído gaussiano em cada *embedding* da entrada, tirando a média dos mapas de saliência calculados por meio do gradiente padrão.

3.2 Técnicas Próprias para o BERT

Desde o surgimento do Transformer, diversas visualizações específicas para esta arquitetura foram desenvolvidas. Elas podem ser divididas em dois grandes grupos: visualizações voltadas para o mecanismo de atenção, e visualizações sobre as representações internas geradas pelos modelos.

3.2.1 Visualizações do Mecanismo de Atenção

Um dos primeiros trabalhos de visualização sobre o Transformers foi disponibilizado por Llion Jones[6], um dos co-autores do Transformers, com a finalidade de visualizar o funcionamento do mecanismo de atenção. Diversos trabalhos ((CLARK et al., 2019); (VIG, 2019); (HOOVER; STROBELT; GEHRMANN, 2020)) seguiram o mesmo formato de visualização, utilizando-o para realizar diversas observações sobre o funcionamento do BERT.

Clark et al. (2019) observam que algumas cabeças espalham sua atenção entre todos os tokens da sentença, como a cabeça 1-1 da Figura 22, onde o primeiro valor representa a camada do BERT e o segundo representa a cabeça. Já outras consideram puramente o aspecto posicional, prestando atenção no token anterior ou posterior, como na cabeça 3-1 da Figura 22. Também encontram cabeças onde tokens muito frequentes, como o "[SEP]" e o ponto final, tem sua nova representação calculada a partir de uma atenção distribuída entre todos os tokens da sentença.

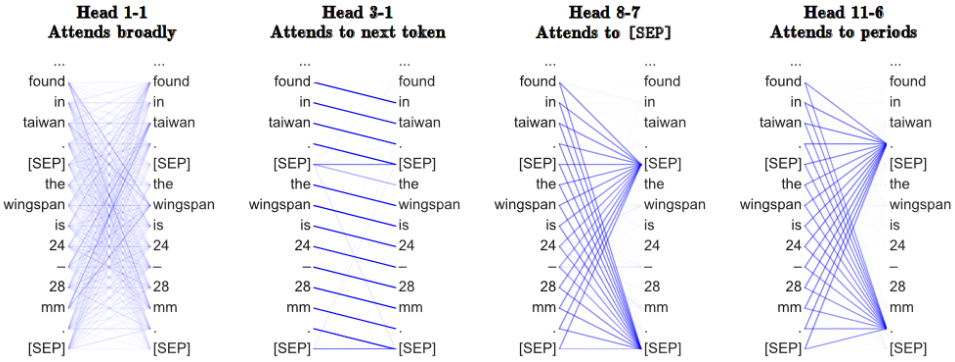


Figura 22 – Padrões de atenção em diferentes cabeças. Figura retirada de (CLARK et al., 2019)

A Figura 23 mostra o uso de duas diferentes cabeças de atenção para efetuar diferentes tarefas gramaticais. À esquerda utiliza-se a cabeça 7-6 como identificador de pronomes possessivos, sem que tenha sido feito um treinamento específico para isso. O BERT utilizado é o disponibilizado após pré-treinamento, sem fine-tuning. Em 80,5% dos casos, o mecanismo de atenção apontou corretamente para os pronomes referentes a substantivos. À direita apresenta-se uma cabeça que 82,5% das vezes pôde apontar corretamente os verbos auxiliares que modificam os verbos investigados. Isso mostra que mesmo que não tenha sido treinado de maneira supervisionada para capturar padrões linguísticos, o BERT aprende por conta própria a identificar e utilizar estes padrões.

Vig (2019) também observam tarefas específicas realizadas por cada cabeça de atenção, exibidas na Figura 24. À esquerda observa-se uma cabeça capaz de observar listas de itens. No centro, uma cabeça que presta atenção em verbos. E na direita uma cabeça que associa nomes compostos a siglas. A ferramenta[7] disponibilizada por (VIG, 2019) é feita para trabalhar tanto

[6] https://github.com/tensorflow/tensor2tensor/tree/master/tensor2tensor/visualization
[7] https://github.com/jessevig/bertviz

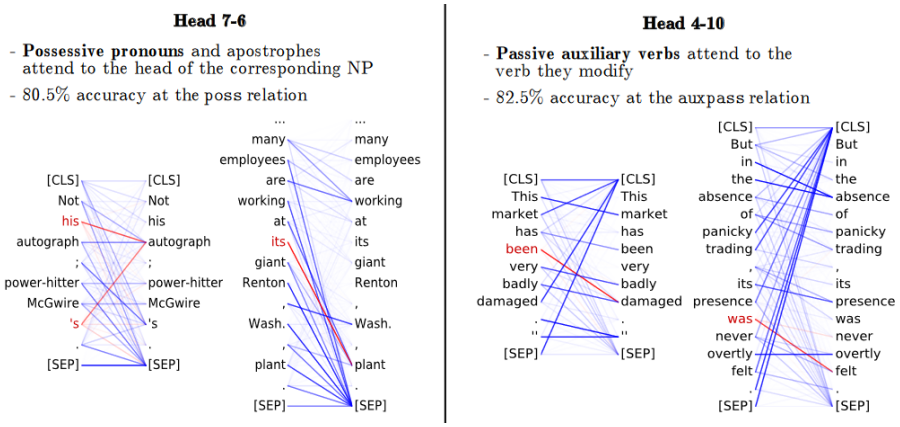


Figura 23 – Observação de padrões linguísticos aprendidos pelo BERT. Figura retirada de (CLARK et al., 2019)

com o BERT, composto por codificadores do Transformer, quanto com o GPT-2, formato por decodificadores do Transformer.

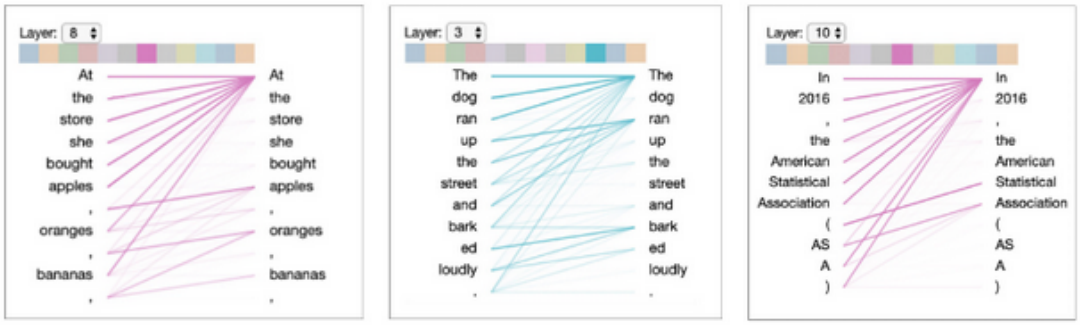


Figura 24 – Observação de padrões linguísticos aprendidos pelo BERT. Figura adaptada de (VIG, 2019)

Uma outra tarefa difícil que o BERT aprende a realizar com suas cabeças é a resolução de correferência, que consiste em achar as várias formas pelas quais uma entidade é representada no texto. A Figura 25 mostra a cabeça 5-4, que realiza esta tarefa com 65,1% de acurácia, onde as linhas em vermelho destacam a correferência.

Para entender o papel dos tokens mais frequentes, Clark et al. (2019) observam a atenção média sobre os tokens "[SEP]", "[CLS]", e o agrupamento entre "." e ",". A Figura 26 mostra a atenção média sobre cada token. Nota-se que o token "[SEP]" é um grande alvo do mecanismo de atenção nas camadas intermediárias.

Clark et al. (2019) argumentam que o token "[SEP]" é usado como no-op, ou seja, para representar uma operação nula. Cada cabeça especializa-se em identificar um ou mais padrões no texto, e quando nenhum padrão é detectado, a rede por padrão foca sua atenção no token "[SEP]". Para corroborar essa hipótese, eles verificam a importância do token "[SEP]" de acordo com o método dos Gradientes Integrados, cujo resultado está apresentado na Figura 27. O token "[SEP]" perde sua importância da quinta camada em diante, local onde passa a chamar mais atenção conforme a Figura 26. Isso mostra que prestar mais ou menos atenção no

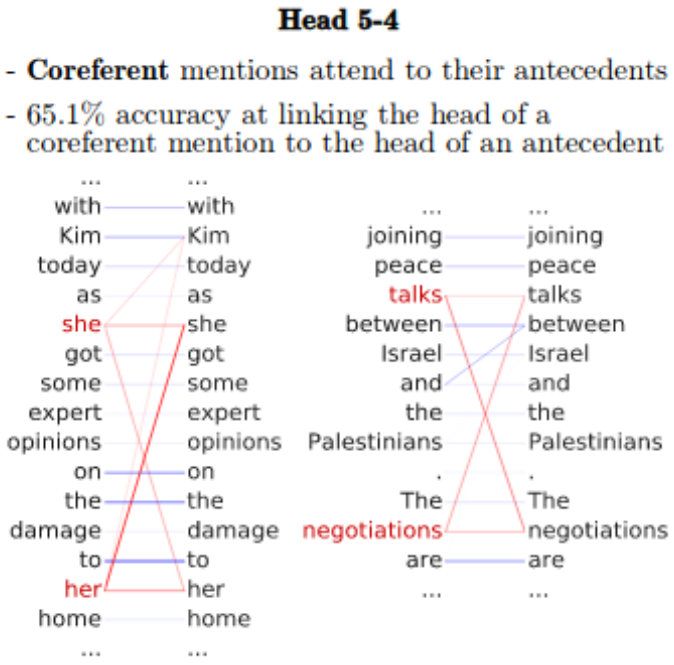


Figura 25 – Cabeça de atenção que efetua resolução de correferência. Figura adaptada de (CLARK et al., 2019)

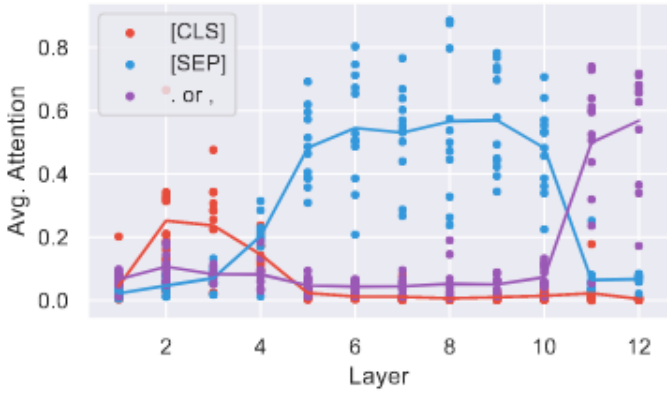


Figura 26 – Atenção média sobre diferentes tokens. Figura adaptada de (CLARK et al., 2019)

token "[SEP]" não afeta a saída do modelo, dando suporte à hipótese de que o token "[SEP]" é usado como no-op.

Um outro aspecto visualizado por Clark et al. (2019) é a tendência das cabeças em cada camada olharem para uma quantidade maior ou menor de tokens. A Figura 28 mostra a entropia na distribuição de atenção em cada cabeça, onde a entropia máxima significa atenção perfeitamente distribuída entre os tokens, e portanto uma alta entropia significa que a cabeça distribui sua atenção entre vários tokens, enquanto uma baixa entropia indica que a cabeça foca em poucos tokens. Observa-se que as camadas intermediárias focam mais sua atenção em uma quantidade pequena de tokens, enquanto as camadas iniciais e finais distribuem mais sua atenção, com destaque para a primeira camada, que quase atinge a entropia máxima.

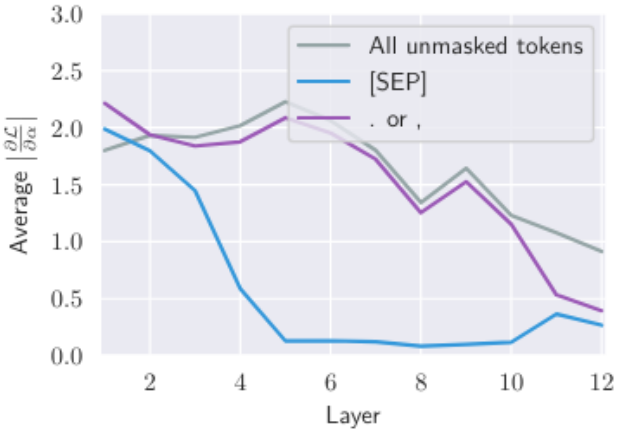


Figura 27 – Importância de diferentes tokens para a tarefa de Modelo de Linguagem Mascarado. Figura adaptada de (CLARK et al., 2019)

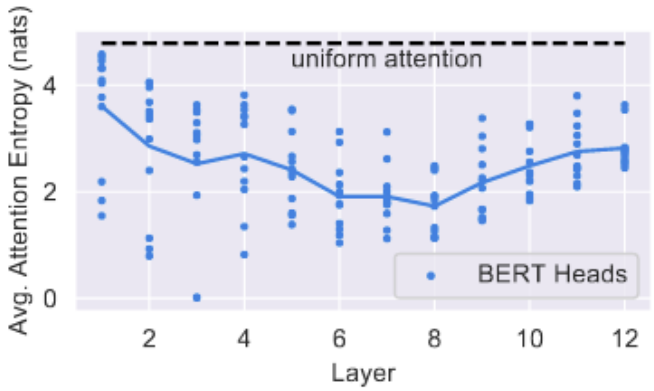


Figura 28 – Entropia na distribuição da atenção em cada cabeça e em cada camada. Figura adaptada de (CLARK et al., 2019)

Clark et al. (2019) ainda apresentam uma técnica para visualizar as cabeças de atenção de acordo com a sua similaridade. Eles calculam a distância entre cada cabeça de acordo com a divergência de Jensen-Shannon (LIN, 1991) entre os tokens de saída da atenção em cada cabeça para cada token. Para uma mesma entrada, toma-se a saída n x k de cada cabeça de atenção[8], e compara-se a divergência, conforme a Equação 20, onde H_i e H_j são duas cabeças distintas.

$$\sum_{token \in data} JS(H_i(token), H_j(token)) \tag{20}$$

A Figura 29, em sua parte superior, mostra o agrupamento obtido após uma projeção multidimensional em duas dimensões que tenta conservar as distâncias entre cada cabeça conforme a divergência de Jensen-Shannon calculada. Na parte inferior destaca-se que muitas cabeças de uma mesma camada acabam sendo muito similares, porém espera-se que uma

[8] No BERT base, cada uma das 12 camadas são compostas por 12 cabeças de atenção, cada uma com dimensão $k = 64$, resultando no total de 768 dimensões para a representação final de cada token, produto da concatenação da saída de cada cabeça.

diversidade de representações seja mais vantajoso para o modelo. Dessa forma é interessante investigar se o *dropout* está causando esta redundância, e se isso é prejudicial para a performance do modelo (CLARK et al., 2019).

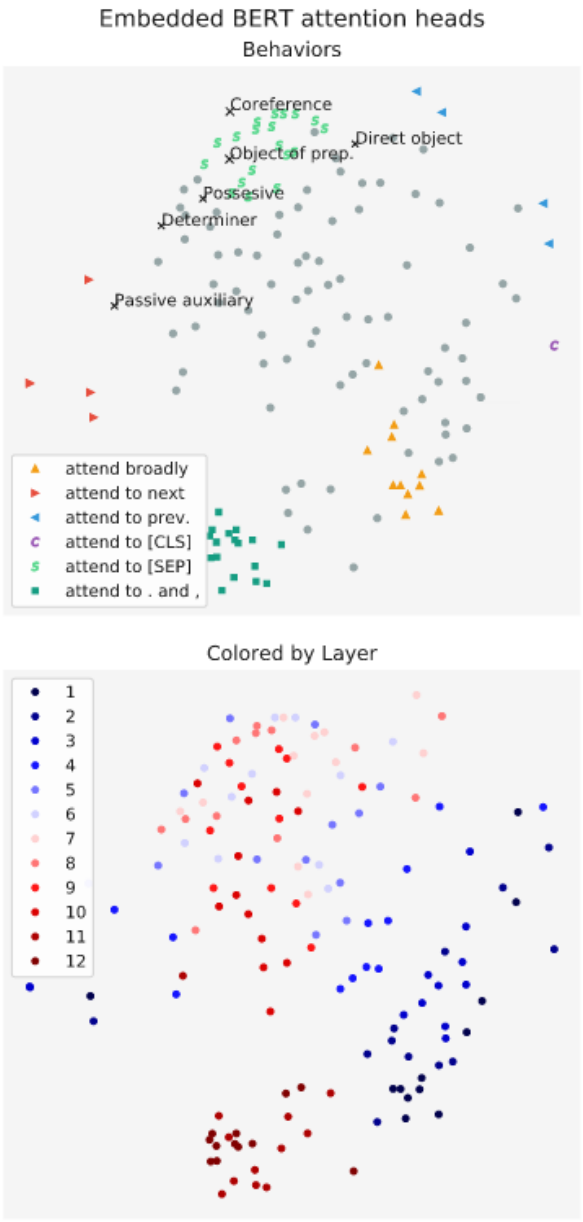


Figura 29 – Agrupamento das cabeças de atenção. Figura adaptada de (CLARK et al., 2019)

Vig (2019) disponibiliza uma visão global do mecanismo de atenção, chamada de Visão do Modelo. Ela permite uma rápida identificação dos diferentes padrões de aplicação de atenção, pois mostra em uma só tela todas as atenções do modelo. A Figura 30 mostra uma parte da visão para o BERT. Notam-se cabeças puramente posicionais, como a 1-1 e 1-4, assim como cabeças que apresentam uma atenção cruzada entre as sentenças A e B, como é o caso da cabeça 3-0, oposição à cabeça 0-0, cujos tokens de uma sentença ignoram os tokens da outra.

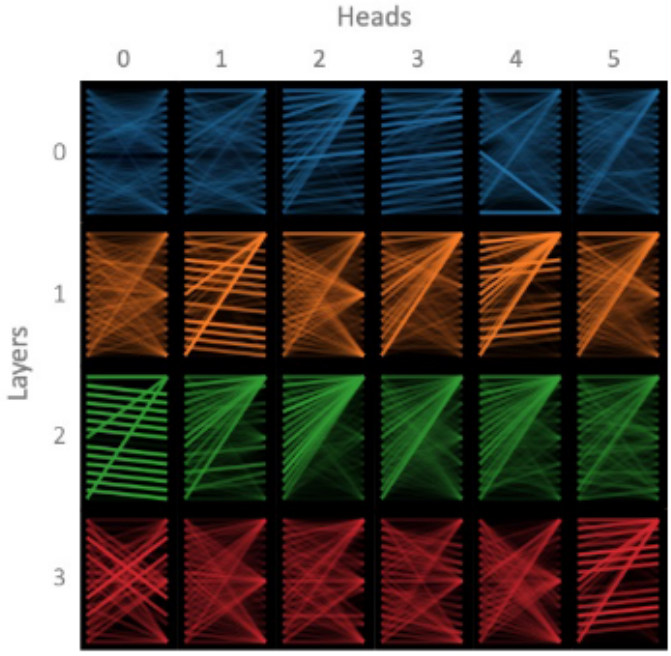


Figura 30 – Visão do Modelo - Figura retirada de (VIG, 2019)

A Figura 30 ainda permite a identificação de cabeças de atenção similares, que podem ser redundantes. Este é o caso das cabeças 2 e 5 da camada 1, assim como das cabeças 1 e 4 da camada 3.

As visualizações do mecanismo de atenção apresentam um problema conforme a rede se aprofunda: a identificabilidade dos tokens. De acordo com o funcionamento do codificador do Transformer, apresentado na Seção 2.2.1, cada representação de token que sai de uma camada é resultado de uma média ponderada pela atenção entre as representações de todos os tokens da entrada da camada. Dessa forma, a representação do token "its" da Figura 23 à esquerda, por exemplo, não contém a semântica apenas do termo "its", sendo uma mistura de várias representações que a rede formou até chegar à quinta camada. Ou seja, cada embedding potencialmente possui informações agregadas de vários tokens. Portanto, as visualizações do mecanismo de atenção não permitem relacionar integralmente uma atenção a um token específico da entrada.

Para mitigar este problema, e fornecer visualizações que identifiquem mais precisamente o token para o qual a atenção é dirigida, Abnar e Zuidema (2020) propõem dois métodos post-hoc para calcular a atenção: attention rollout e attention flow.

Ao invés de considerar apenas os scores de atenção para a formação das representações de saída em cada camada, as conexões residuais também são levadas em conta ao considerar 50% da contribuição como do próprio token e os demais 50% de acordo com o mecanismo de atenção. A Equação 21 mostra o cálculo dos valores da atenção ajustada ao considerar as conexões residuais. W_{att} é a matriz de atenção comum, obtida durante o processo de inferência, e "A"é a matriz de atenção ajustada.

$$A = 0,5W_{att} + 0,5I \tag{21}$$

Attention Rollout

A atenção em cada camada é calculada conforme a Equação 21, e então para descobrir a atenção sobre cada token da entrada, multiplica-se a nova atenção de maneira recursiva, iniciando pela camada de entrada, conforme a Equação 22. Dessa forma, cada camada leva em conta a atenção das camadas anteriores para ajustar a atenção que foi prestada em cada token, aumentando a identificabilidade dos tokens nas camadas mais profundas. Para a primeira camada, a nova atenção ($\tilde{\text{A}}$) mantém-se a mesma, mas nas demais camadas a matriz de atenção será ajustada pela multiplicação com a matriz de atenção ajustada da camada imediatamente anterior.

$$\tilde{\text{A}}(l_i) = \begin{cases} A(l_i)\tilde{\text{A}}(l_{i-1}), & \text{se } i > j \\ A(l_i), & \text{se } i = j \end{cases} \tag{22}$$

Attention Flow

Esta técnica modela o mecanismo de atenção como uma rede de fluxo, estrutura advinda da teoria dos grafos. A aplicação de um algoritmo de fluxo máximo provê uma aproximação da atenção de qualquer token, em qualquer camada (vértice final), em relação a qualquer token da entrada (vértice inicial).

Avaliação

Abnar e Zuidema (2020) comparam as atenções calculadas contra a importância de cada token, dada pelo método blank-out, apresentado na Seção 3.1.1. Então os autores observam a correlação entre os scores do blank-out e a importância dada pelos três métodos propostos. O attention flow mostrou-se um pouco melhor que o attention rollout, que por sua vez mostrou-se muito melhor que a atenção pura. A exceção é a primeira camada, onde a atenção pura está mais correlacionada à importância de cada token. A correlação dada pelos gradientes da entrada, ao invés do blank-out, também mostraram a mesma tendência.

Para finalizar esta Seção sobre visualizações do mecanismo de atenção, temos a Visão do Neurônio, também apresentada por Vig (2019). Esta visualização mostra em detalhes o cálculo da atenção entre cada par de tokens, ao nível de cada dimensão das representações. Selecionado um token da entrada, mostra-se o cálculo da similaridade entre o vetor query deste token e o vetor key de todos os demais, conforme a Figura 31, que mostra a visão do neurônio para a cabeça 0 da camada 0. Como sugere a Figura 30, os tokens da sentença A são levados muito mais em consideração do que os da sentença B.

Um comportamento destacado por Vig (2019) ao utilizar a Visão do Neurônio é de uma cabeça de atenção que funciona como uma janela de contexto, levando em maior consideração

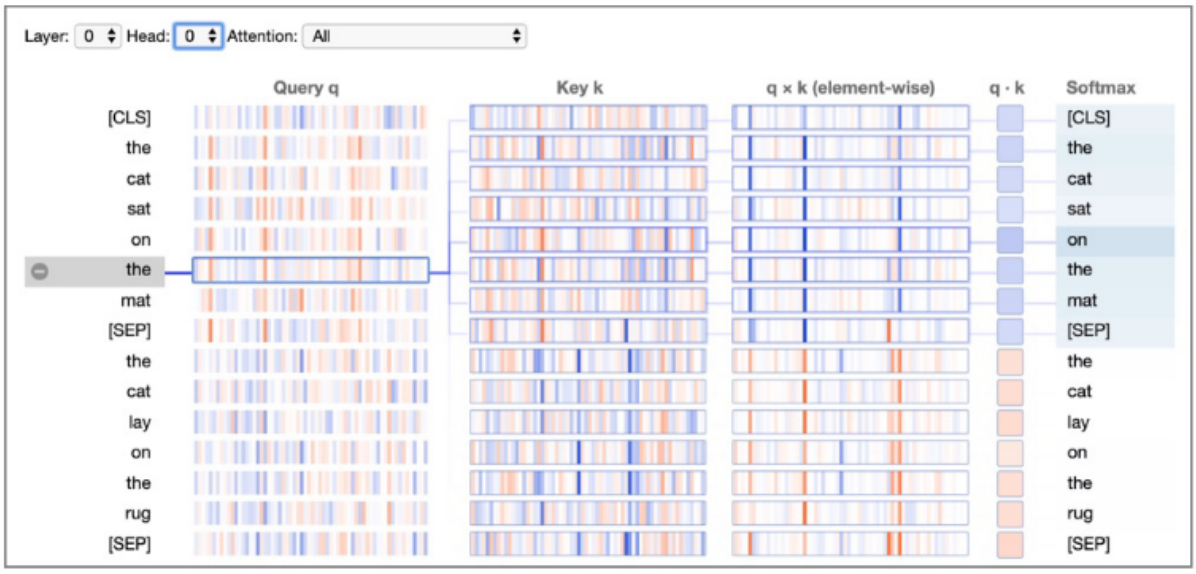


Figura 31 – Visão do Neurônio - Retirada de (VIG, 2019)

os tokens mais próximos, como consta na Figura 32. Para construir a nova representação do ponto final, utiliza-se cada vez menos os demais tokens conforme sua distância.

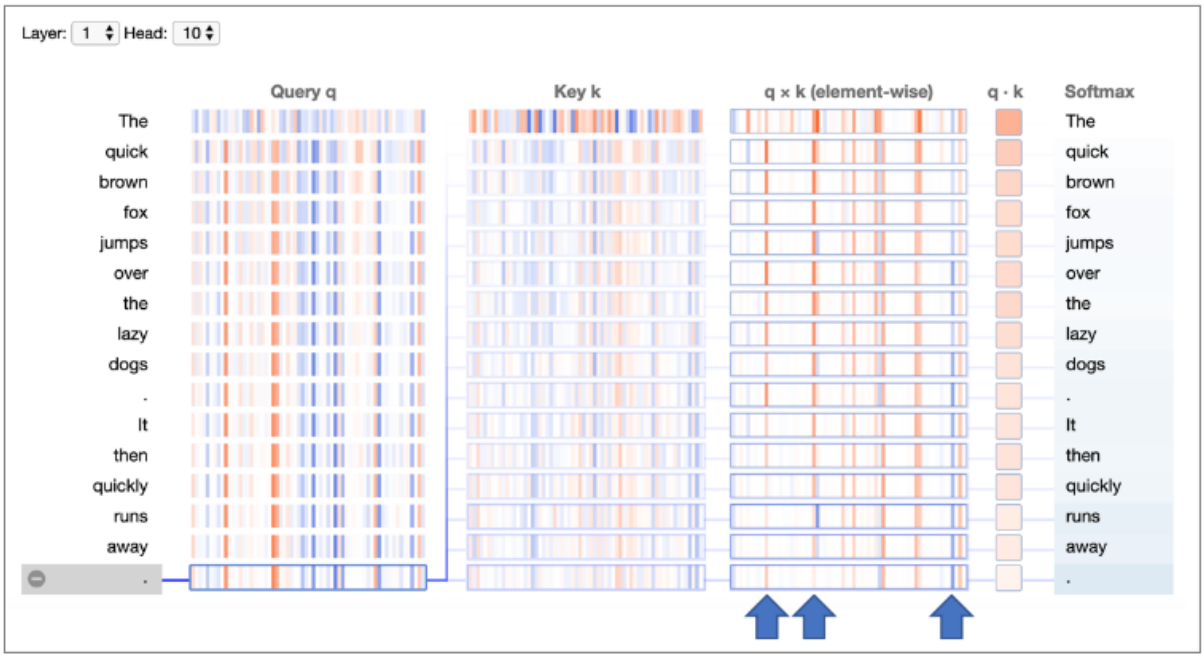


Figura 32 – Visão do Neurônio - Retirada de (VIG, 2019)

3.2.2 Visualizações das Representações Internas

3.2.2.1 Projeção Multidimensional

A VisBERT (AKEN et al., 2020) é uma ferramenta direcionada para o entendimento do BERT por meio da análise das representações internas construídas por cada camada, portanto é complementar às visualizações dos mecanismos de atenção. Dada uma instância que será

investigada, tomam-se as representações que saem de cada bloco codificador, e então aplica-se a técnica PCA (Principal Component Analysis) para projetar os vetores em duas dimensões, tornando-os aptos à visualização humana. Então cada camada é visualizada separadamente.

A ideia da visualização é que quanto mais próximos os tokens estiverem, mais similar a semântica entre eles. Aken et al. (2020) conseguiram identificar que o BERT passa por quatro fases ao realizar a tarefa de question-answering: i) clusterização de tópicos; ii) conexões de entidades à menções e atributos; iii) pareamento das questões com fatos de suporte; e iv) extração da resposta.

Segue um exemplo do uso da VisBERT[9] para responder à pergunta "Does Darwin prefere medicine or marine biology?", cuja resposta encontra-se no trecho "Darwin's early interest in nature led him to neglect his medical education at the University of Edinburgh; instead, he helped to investigate marine invertebrates.", retirado da Wikipedia. A resposta esperada é "marine biology", que o modelo encontra corretamente. Durante o processo de inferência, atribui-se às camadas 1 e 2 a fase de clusterização de tópicos; às camadas 4 e 5 a fase de conexão de entidades à menções e atributos; às camadas 7 e 8 a fase de pareamento da questão com fatos de suporte; e às camadas 10, 11 e 12 a fase de extração da resposta. As demais camadas são consideradas de transição.

A Figura 33 mostra a clusterização de tópicos. Os tokens "Darwin" e "biologia" estão próximos, assim como medicina e medicinal. Os tokens são coloridos da seguinte maneira: os tokens da questão estão em azul escuro; os tokens roxos compõem a resposta final; e os demais tokens são do texto - em azul claro, com os tokens de suporte destacados em verde.

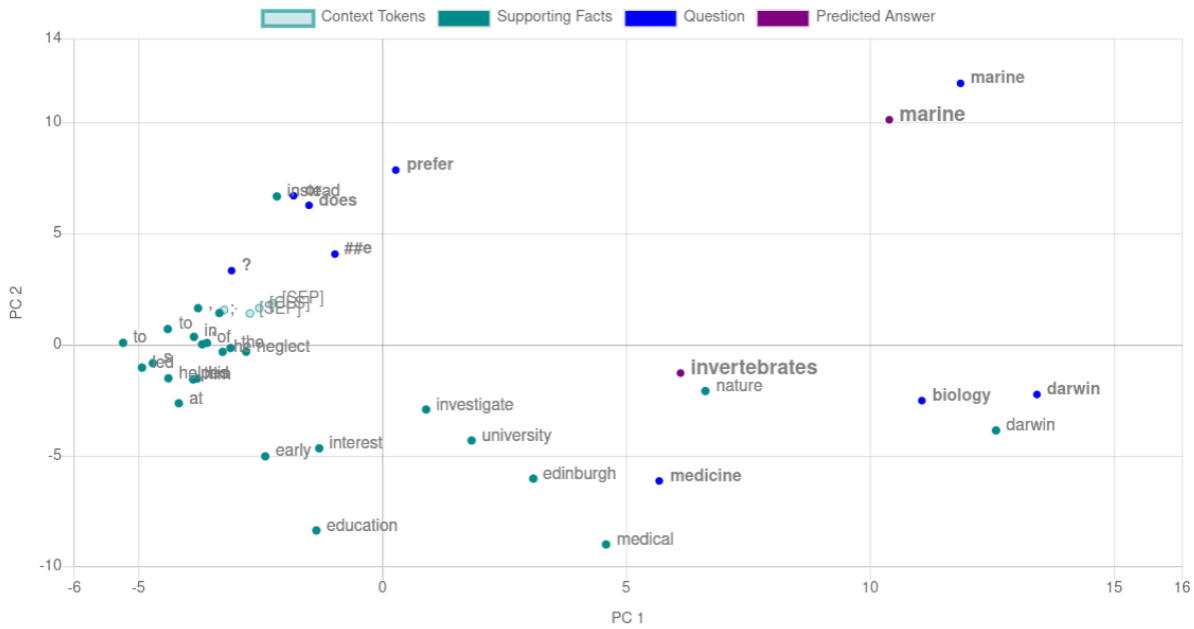


Figura 33 – Fase de clusterização de tópicos

[9] https://visbert.demo.datexis.com/

A Figura 34 mostra a conexão de entidades com menções e atributos. Observa-se como o adjetivo "marine" se aproximou de "invertebrados" e "biologia", que são modificados por ele.

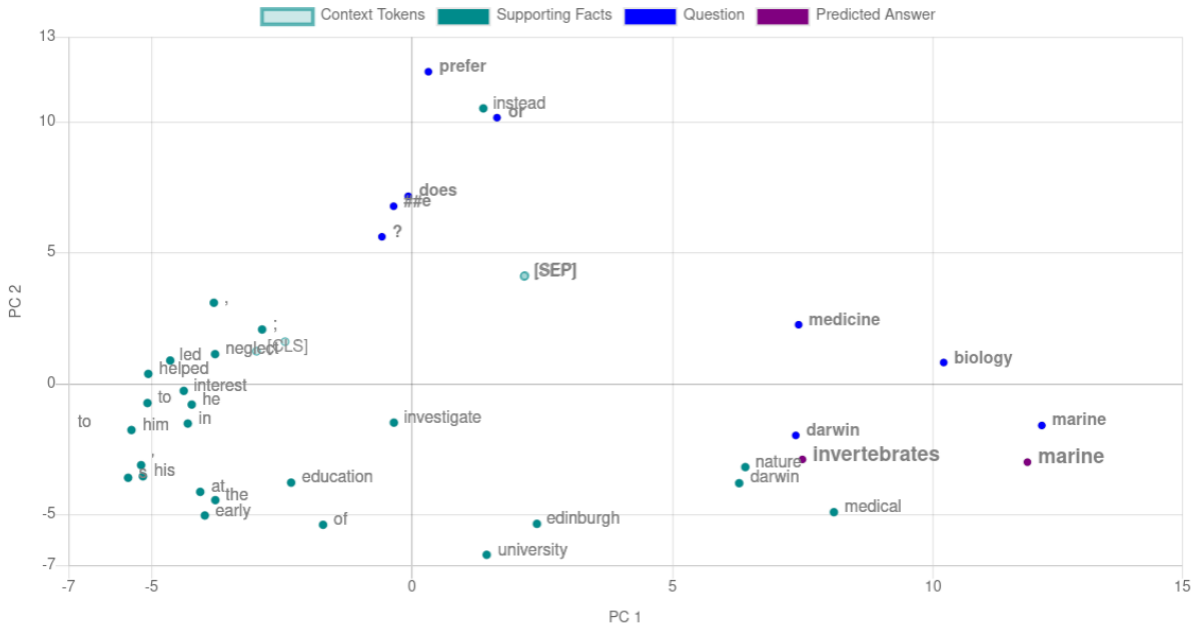


Figura 34 – Fase de conexão entre entidades e suas menções e atributos

Esta visualização torna-se importante para verificar a evolução entre as representações em cada camada, fornecendo mais uma ferramenta para a verificação de erros.

3.2.2.2 Similaridade entre Entradas

Outra forma de analisar as representações internas do BERT é, dado um exemplo de entrada, encontrar outros exemplos no dataset com representações similares. A ferramenta EXBERT, apresentada por Hoover, Strobelt e Gehrmann (2020), além de fornecer a visualização padrão para os mecanismos de atenção, também busca exemplos similares ao que está sendo investigado, para assim elucidar melhor ao usuário o funcionamento do modelo. Esta característica de incorporar o dataset às técnicas de interpretabilidade não é tão frequente na literatura, porém é de extrema importância para obter-se melhores explicações (BRASOVEANU; ANDONIE, 2020).

A busca por similaridade do EXBERT visa encontrar exemplos tomando como ponto de partida um token em uma camada específica, mostrando 50 exemplos do dataset que contém um token com representação similar na mesma camada, ordenados de acordo com a similaridade cossenoidal. Duas maneiras de representar um token são permitidas pela ferramenta: busca por embedding e por contexto.

Na busca por embedding, a similaridade é tomada entre as representações de cada token considerando a representação de saída em cada camada. Já a busca por contexto considera como representação de cada token a concatenação da saída das cabeças selecionadas. Ou seja, a busca por embedding mostra os exemplos mais similares considerando a semântica geral de

uma camada, enquanto a busca por contexto mostra os exemplos mais similares de acordo com uma cabeça específica, ou um conjunto de cabeças, permitindo que o usuário explore muito mais o modelo.

3.3 Qualidade em conjuntos de dados rotulados

Ao longo da última década o foco principal no treinamento de redes neurais tem sido no desenvolvimento de melhores arquiteturas e estratégias de treinamento, tendo como entrada datasets fixos durante todo o processo; um paradigma chamado de *Model-Centric AI*, e que levou a grandes avanços na área de *deep learning*. Atualmente tem-se identificado a importância de dar mais atenção ao dado de entrada, possibilitando maiores ganhos ao manter-se a rede neural fixa, mas continuamente melhorar a qualidade dos dados de treinamento; um paradigma chamado de *Data-Centric AI*. Trabalhar com anotações criteriosas em casos cuidadosamente selecionados permite atingir bons resultados com um número pequeno de exemplos, em oposição ao uso de datasets volumosos onde uma grande quantidade de exemplos é necessária para superar os prejuízos causados por ruídos nos dados (STRICKLAND, 2022).

Melhorar a qualidade de conjuntos de dados não é algo novo, pois muitas aplicações comerciais dependem de dados de qualidade. Polyzotis e Zaharia (2021) discutem diversas estratégias existentes nas áreas de engenharia de dados e engenharia de *machine learning* que podem beneficiar na criação e manutenção de datasets de qualidade. Entre elas, destaca-se a necessidade de continuamente monitorar os modelos em produção e atualizar os dados de treinamento quando detectar-se uma discrepância com os dados do mundo real. Bloquear conteúdos em redes sociais, por exemplo, é uma tarefa muito dinâmica pois novos assuntos surgem a todo momento. Dessa forma, modelos para esta finalidade devem ser atualizados diariamente ou com uma frequência ainda maior.

A seguir são apresentados dois trabalhos com a finalidade de melhorar a qualidade de anotações. O primeiro apresenta um processo para a filtragem dos melhores casos para treinar um modelo, onde a remoção de um terço dos exemplos acarretou em um ganho de 5% no desempenho. O segundo discute o uso de modelos simples para auxiliar na análise dos datasets, permitindo a identificação dos casos mais difíceis de classificar, passando pela detecção de casos mal rotulados. Por fim, apresenta-se uma forma de identificar e mitigar vieses não-intencionais incorporados pelos modelos durante o seu treinamento.

3.3.1 Construção de datasets menores e melhores

Motamedi, Sakharnykh e Kaldewey (2021) propõem um pipeline para a construção de um dataset com maior qualidade, partindo de um dataset já rotulado. A ideia é iniciar com uma pequena quantidade de exemplos selecionados aleatoriamente e revisados por um humano para garantir a corretude das anotações, e então ir incrementando o dataset com

novos casos. O pipeline é composto por cinco passos: i) eliminação de casos duplicados para eliminar redundâncias e garantir que o conjunto de treinamento não possui casos que estão no conjunto de validação; ii) treinamento de modelos auxiliares para a seleção de novos exemplos; iii) uso dos modelos auxiliares para a seleção de casos para revisão; iv) inclusão de novos casos para compensar o desbalanceamento; e v) validação cruzada com n-folds para avaliação do modelo obtido.

Os passos ii e iii formam o loop principal para a qualidade dos dados. Partindo de um conjunto de exemplos pequeno, mas perfeitamente rotulado, um modelo auxiliar é treinado e usado para inferência em todo o restante do dataset. Motamedi, Sakharnykh e Kaldewey (2021) então selecionam os 500 casos com menor erro de classificação, medido pela função de loss do modelo, que em teoria são casos corretamente rotulados e característicos das classes. Também são selecionados os 100 casos com maior erro de classificação, que podem ser casos mal rotulados – e por isso apresentam loss elevada – que devem ser corrigidos, exemplos ruins que devem ser descartados do dataset, ou casos que simplesmente são difíceis e que devem continuar no conjunto. Todos os casos são revisados por humanos e incluídos no dataset limpo, e então uma nova rodada se inicia: treinamento de um modelo auxiliar, seguido pela seleção e revisão de 600 novos casos, e inclusão no dataset limpo. Motamedi, Sakharnykh e Kaldewey (2021) efetuam o procedimento em todo o dataset em seu experimento, não discutindo qual seria o ponto de parada caso apenas um subconjunto do dataset fosse revisado.

Aplicado todo o procedimento em um dataset com algarismos romanos, que originalmente possui 2067 exemplos de treinamento, e após a limpeza fica com 1370 exemplos, a acurácia – medida no mesmo conjunto de teste – subiu de 64% para 69%. Ou seja, um dataset menor e com maior qualidade é capaz de gerar um modelo melhor.

3.3.2 Uso de modelos simples para análise das anotações

Paiva et al. (2021) apresentam a ferramenta PyHard[10], que permite a inspeção de datasets conforme diferentes características. Diferentes modelos simples, como Regressão Logística, SVMs e MLPs pequenas são treinados em um regime de validação cruzada, e utilizados para tomar a probabilidade de cada instância em relação ao seu rótulo, que é subtraída de 1. Assim, quanto maior o valor, maior o erro. A média entre todos os modelos é o resultado final, chamado de *Instance Hardness*, uma medida que indica a dificuldade de se classificar cada instância do dataset. Os casos em que o erro é muito grande podem ser realmente casos difíceis, mas também podem ser erros de rotulação.

Os autores também utilizam diferentes algoritmos para determinar medidas de dificuldade em classificar determinadas instâncias, chamadas de *Hardness Measures*, como clusterizações que possuem casos de classes distintas muito próximas, e utilizam estas informações em conjunto com a *Instance Hardness* para gerar meta-características que então são

[10] https://pypi.org/project/pyhard/

reduzidas para duas dimensões a fim de permitir a visualização dos casos do dataset. Estes algoritmos estão fora do escopo deste documento, pois apenas pela *Instance Hardness* já é possível identificar casos que necessitam de revisão humana.

A Figura 35 mostra uma visualização resultante do experimento realizado por (PAIVA et al., 2021), onde a cor vermelha representa os casos com maior dificuldade de classificação, e os azuis são os mais fáceis. Em conjunto com a análise das *Hardness Measures*, que não são apresentadas, os autores concluíram que os agrupamentos destacados contém baixa probabilidade de pertencer à classe de sua rotulação, e possuem elementos de classes distintas muito próximos. Dessa forma torna-se fácil a inspeção de outliers e casos mal rotulados.

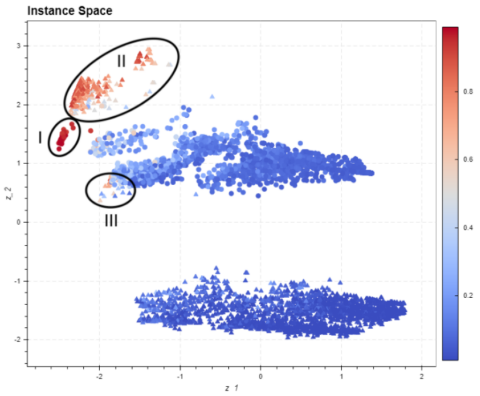


Figura 35 – Visualização do dataset colorido pela dificuldade de classificação. Figura retirada de (PAIVA et al., 2021)

3.3.3 Vieses Não-Intencionais

Todos os modelos de aprendizado de máquina utilizam-se de vieses para tomar suas decisões. Este vieses podem ser intencionais, como um detector de comentários tóxicos apoiar-se em xingamentos para aumentar a probabilidade de classificação, ou não-intencionais caso este mesmo detector atribua alta probabilidade de toxicidade para textos contendo o termo "gay" usado em um contexto de sentimento neutro, por exemplo (DIXON et al., 2018). Os vieses não-intencionais são um reflexo dos dados de treinamento, e devem ser tratados para garantir que os modelos de IA sejam justos em suas decisões. No presente trabalho, a mitigação dos vieses é analisada com a intenção de tornar o modelo mais robusto ao apoiar-se em uma maior diversidade de características.

Para identificar os vieses não-intencionais, (DIXON et al., 2018) preparam templates na forma de frases onde diferentes termos são utilizados, verificando se o modelo considera as sentenças tóxicas ou não. Seja a frase "ele é <token>" um template, pode-se ter as variações "ele é homem" e "ele é gay". Um modelo justo consideraria ambas as frases como não-tóxicas, mas ao considerar "ele é gay" como tóxico, e "ele é homem" como não-tóxica, identifica-se um viés não-intencional relacionada ao termo "gay".

Para avaliar os vieses não-intencionais de forma quantitativa, (DIXON et al., 2018) calculam a taxa de falsos positivos (FPR, de False Positive Rate) e a taxa de falsos negativos (FNR, de False Negative Rate) para cada termo, isto é, no subconjunto de textos que contém o termo, comparando com a FPR e FNR do dataset completo. A taxa de falsos positivos é calculada conforme a Equação 23, e a de falsos negativos conforme a Equação 24, onde "FP" é a quantidade de casos falsos positivos, "TN" os verdadeiros negativos, "FN" os falsos negativos, e "TP" são os verdadeiros positivos.

$$FPR = FP/(FP + TN) \tag{23}$$

$$FNR = FN/(FN + TP) \tag{24}$$

Por fim, para analisar se um termo corresponde ao viés do modelo, compara-se as taxas de falsos positivos e falsos negativos do termo com as taxas no dataset completo. Isso é feito pois se o termo apresentar 10% de falsos positivos, mas o modelo como um todo também apresentar 10% de falsos positivos, o erro não pode ser atribuído ao termo.

Para reduzir os vieses não-intencionais, (DIXON et al., 2018) buscam equalizar a proporção dos termos de interesse no dataset de treinamento, mantendo a mesma proporção nos comentários tóxicos e não-tóxicos. No exemplo apresentado pelos autores, a palavra "gay", por exemplo, foi encontrada em 3% dos comentários rotulados como tóxicos, mas em apenas 0,5% dos não-tóxicos. A solução apresentada é adicionar textos não-tóxicos contendo a palavra "gay" no dataset de treinamento, de tal forma que 3% dos comentário não-tóxicos a contenham.

4 Mapa de Instâncias

Métricas como acurácia, precisão, recall e f1-score são as formas mais comuns para avaliar a performance de um classificador, juntamente com a matriz de confusão, que agrega as classificações e contabiliza os erros de predição cometidos pelo modelo (FERRI; HERNáNDEZ-ORALLO; MODROIU, 2009). Mas além destas métricas, que permitem avaliar o modelo de forma quantitativa, a análise qualitativa é importante para entender o que acontece ao nível das instâncias.

Apresenta-se aqui uma ferramenta desenvolvida para apoiar a análise de classificadores, facilitando a realização da análise qualitativa, chamada de Mapa de Instâncias. Trata-se de uma visualização orientada a pixels, útil para representar grandes quantidades de dados em pequenos espaços, o que fazem por meio do uso de mapeamentos de cores e padrões de pixels (KEIM, 2000). Assim, em um cenário cujo objetivo é analisar as predições de um classificador, este tipo de visualização foi empregado para representar o conjunto de teste inteiro de maneira eficiente. Mais especificamente, no presente trabalho o Mapa de Instâncias foi avaliado como uma ferramenta para a revisão de anotações, ajudando a encontrar textos mal rotulados, assim como para a qualidade geral do dataset, permitindo a identificação de textos mal escritos.

As visualizações orientadas a pixel permitem uma densa disponibilização de informações pois cada pixel é um ponto de informação. Para comparação: um caractere na tela, com fonte tamanho 11, ocupa 100 pixels. Ou seja, no espaço comumente ocupado por um caractere é possível colocar 100 instâncias de um dataset. Em nossa implementação cada instância é apresentada como um quadrado 10x10, o que reduziu a quantidade de informação por pixel na tela, mas permitiu uma interatividade mais fluida com a ferramenta para que o analista possa utilizar o mouse e clicar nos quadrados que representam as instâncias, explorando mais facilmente cada instância em detalhes.

Para demonstração da ferramenta, assim como para a realização de experimentos, utilizamos o BERT treinado para classificação de textos no dataset Yahoo Answers. Mais detalhes sobre o dataset, assim como o processo de treinamento e avaliação do modelo encontram-se no Apêndice A.

Na Seção 4.1 o mapa de instâncias é apresentado com mais detalhes, enquanto a Seção 4.2 descreve a aplicação de um streamgraph (BYRON; WATTENBERG, 2008) para a visualização das predições ao longo do treinamento do modelo. Um experimento para avaliar a utilidade da ferramenta como um revisor de anotações, que mostrou-se muito eficiente para esta finalidade, é descrito na Seção 4.3.

4.1 Detalhamento da Ferramenta Mapa de Instâncias

O mapa de instâncias permite que o usuário veja todo o seu dataset de forma compacta, muitas vezes em uma só tela, a depender do seu tamanho. O mapa representa o dataset na forma de um retângulo composto por vários quadrados pequenos, dispostos lado a lado, como mostra a Figura 36. Cada quadrado representa uma instância, neste caso coloridas de acordo com a classe predita em cada uma, mas agrupadas de acordo com os rótulos. Dessa forma é possível identificar rapidamente os erros do dataset, pois os casos corretamente classificados formam uma área monocromática, evidenciando as instâncias com cores diferentes. A Figura 36 mostra também a legenda de cores empregada para as classes, que seguirá a mesma para todas as demais figuras neste documento.

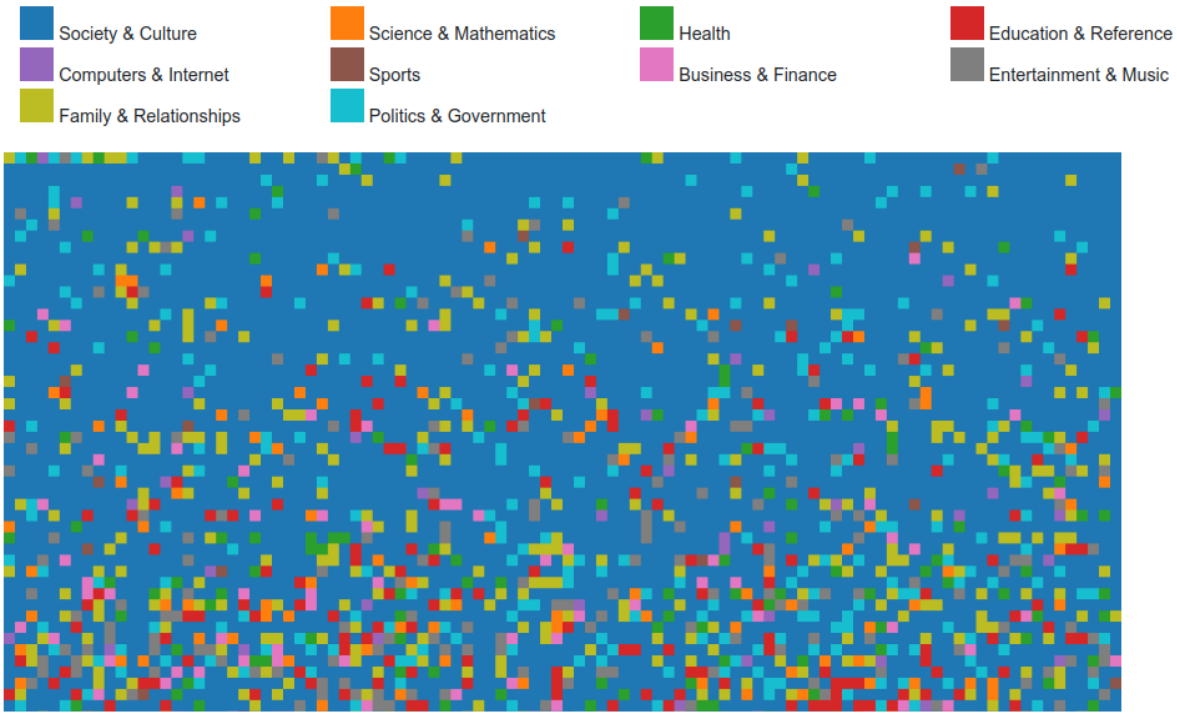


Figura 36 – Tela do mapa de instâncias carregado com o dataset Yahoo Answers

A Figura 36 foi recortada da tela da ferramenta, onde a largura do mapa é limitada a mil pixels, exibindo portanto cem instâncias por linha, e a altura é de 500 pixels, possibilitando ao total a visualização de 5 mil instâncias ao mesmo tempo. Estas configurações são específicas da ferramenta desenvolvida, pois ainda há espaço para legendas, preenchimentos de página, e outras questões de layout, mas em teoria o mapa pode ocupar a tela inteira, que no caso de um monitor de 1600x900 permitiria a visualização de 14400 instâncias ao mesmo tempo.

A Figura 37 mostra uma colagem do mapa completo, onde a cor de fundo de cada região destaca o rótulo das instâncias. Ou seja, a regisão mais acima e à esquerda contém as instâncias rotuladas como “Society & Culture”, a região abaixo, em laranja, contém as instâncias rotuladas como “Science & Mathematics”, etc. Na ferramenta a visualização consiste

nos mapas de cada classe empilhados, mas para adequar ao formato deste documento os mapas são exibidos em duas colunas. O conjunto de teste do dataset Yahoo Answers possui 60 mil exemplos, e como cada instância é representada por um quadrado de 10x10 pixels, seria necessária uma tela um pouco maior do que 2800 x 2100 para exibí-lo completo em uma só tela.

Uma característica importante do mapa de instâncias é a ordenação das instâncias dentro de cada agrupamento, que no presente caso são ordenadas de acordo com a probabilidade que o classificador deu para a classe predita. Por exemplo, a primeira instância do mapa, ou seja, a primeira à esquerda e acima, é um texto rotulado como "Society & Culture" que foi classificado como "Family & Relationships" com 99,50% de probabilidade. Este exemplo trata-se de um erro de anotação, pois o texto fala sobre um término de relacionamento e dúvidas sobre reatar o namoro ou não. A instância à sua direita é a que possui segunda maior probabilidade de classificação dentre os textos rotulados como "Society & Culture", e assim por diante. Ao chegar ao final da linha, as instâncias são dispostas a partir do início da segunda, até termos a instância com menor probabilidade na última linha e na posição mais à direita.

Devido à ordenação pela probabilidade na predição, os casos no topo de cada classe são aqueles nos quais o modelo está muito confiante, enquanto os exemplos de baixo ocorrem com pouca confiança. Dessa forma fica fácil encontrar e analisar os erros mais críticos, que ocorrem com probabilidade muito alta. Esta foi a forma escolhida no presente trabalho, que resultou em uma boa forma de identificar-se erros de rotulação e textos mal escritos, contudo, o analista pode usar qualquer forma de ranqueamento para ordenar as instâncias do dataset, a depender do objetivo de sua análise.

4.2 Distribuição de probabilidades ao longo do treinamento

A ferramenta desenvolvida permite que o usuário clique no quadrado representante de uma instância e observe detalhes sobre a mesma. Além do texto e o rótulo, também são mostradas a classe predita, a probabilidade da predição, e uma segunda visualização na forma de um streamgraph (BYRON; WATTENBERG, 2008). A Figura 38 mostra um caso de "Society & Culture" que o modelo entende também como "Science & Mathematics". O texto mostra o título da pergunta, seu conteúdo, e a resposta separados por linhas em branco.

O streamgraph é usado para mostrar a classificação que o modelo dá para o texto ao longo do seu ciclo de treinamento. O eixo X corresponde às épocas do treinamento. No presente estudo, um checkpoint do classificador foi salvo seis vezes durante o treinamento de cada época, e então fez-se a inferência com cada checkpoint. No eixo Y, cada classe é representada por uma cor diferente e a altura da faixa de cada cor corresponde à probabilidade que o modelo dá para cada classe, somando 100%. Dessa forma é possível visualizar como as probabilidades de classificação foram alterando durante o treinamento. Na Figura 38 o

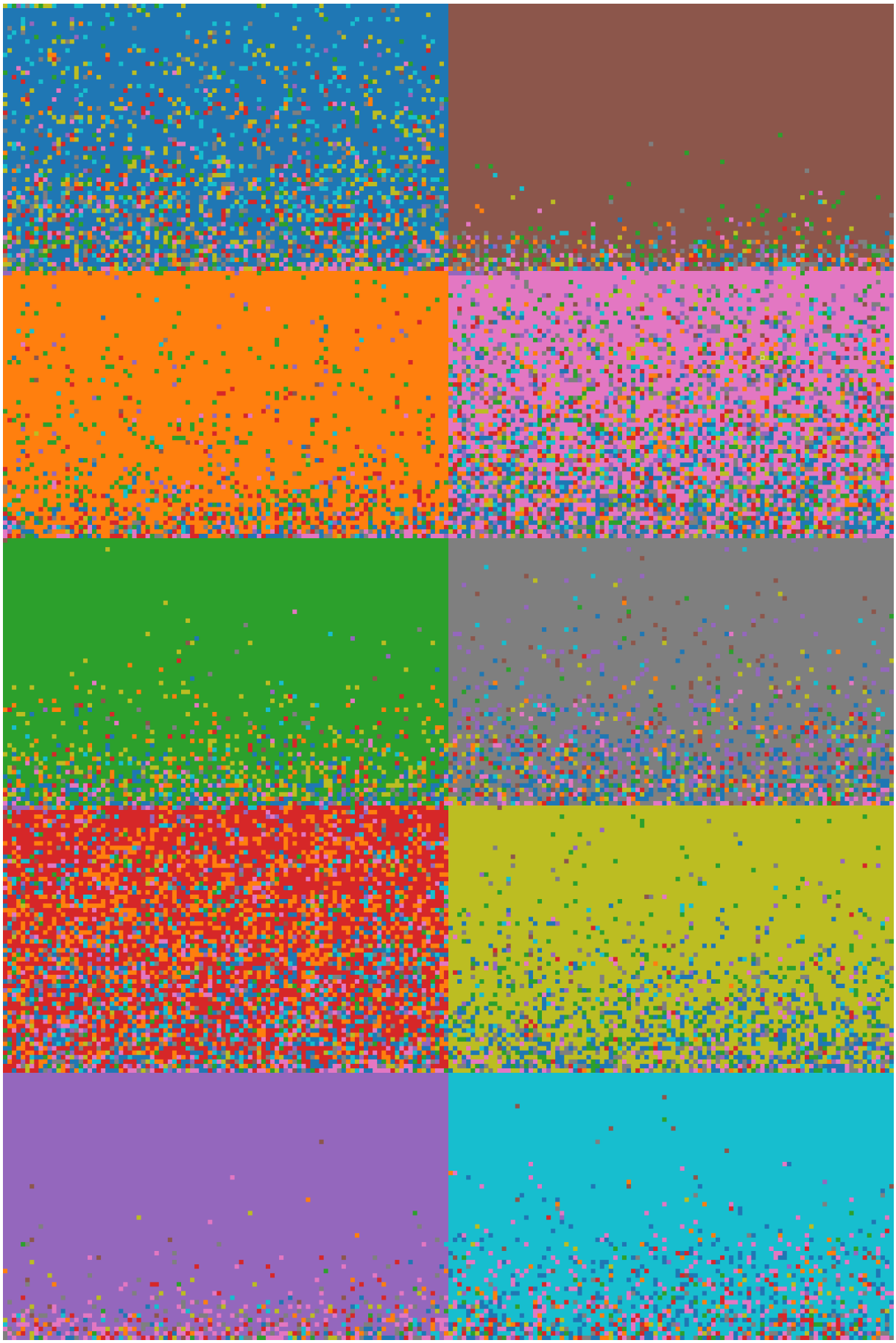


Figura 37 – Tela do mapa de instâncias carregado com o dataset Yahoo Answers

streamgraph mostra como o classificador oscila a sua decisão ao longo do treinamento, mas é notável a tendência para classificar como "Society & Culture" com mais treinamento.

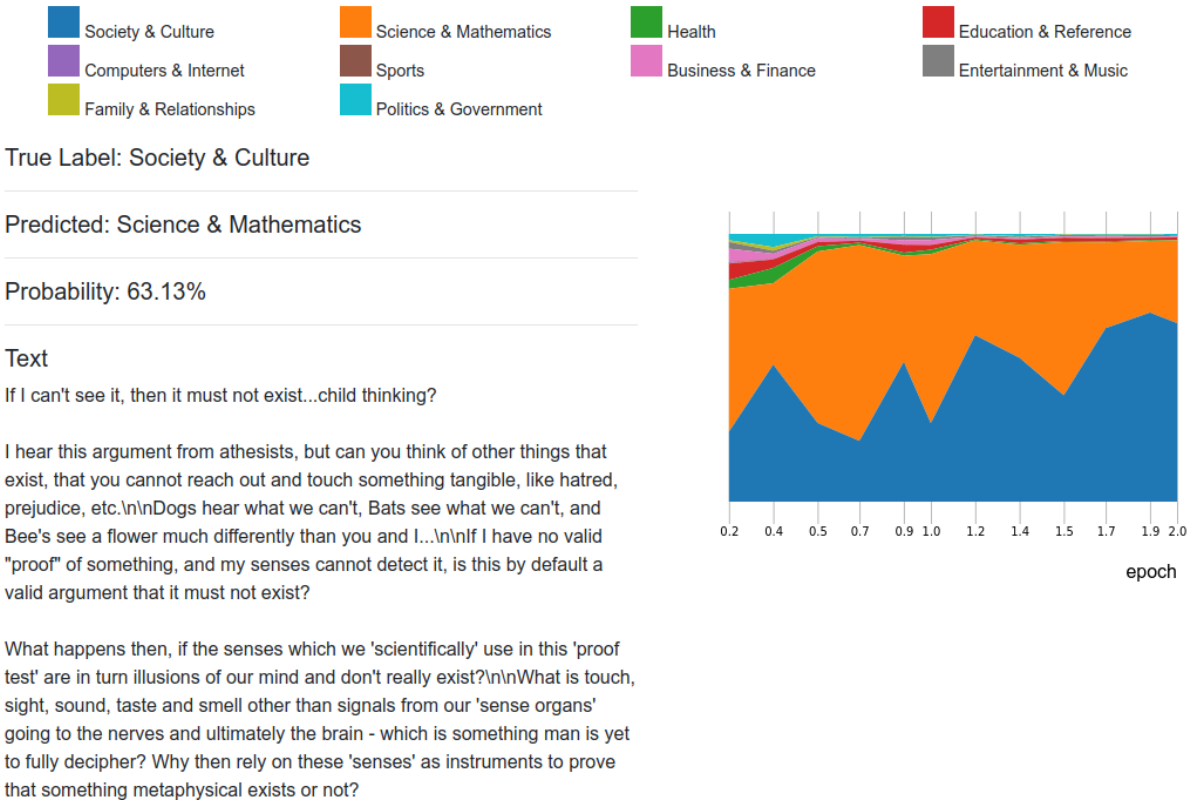


Figura 38 – Detalhes de uma instância rotulada como Society & Culture, que ao longo do treinamento o modelo deixa de classificar como Science & Mathematics

A Figura 39 mostra os detalhes da instância do topo, citada anteriormente, onde podemos ver o streamgraph praticamente todo amarelo. Ou seja, a probabilidade para "Family & Relationships" manteve-se alta durante todo o processo, praticamente sem espaço para as outras classes.

Outro caso interessante que o streamgraph ajuda a identificar são os textos ambíguos, que na verdade pertencem a mais de uma classe, ainda que o dataset não seja multi-rótulos. A Figura 40 mostra os detalhes de uma instância que está rotulada como "Computers & Internet", mas que também deveria estar associada à "Politics & Government".

4.3 Experimento: Uso da ferramenta como revisor de anotações

Para avaliar a capacidade do Mapa de Instâncias na identificação de casos mal rotulados, juntamente com o método escolhido para a ordenação das instâncias, efetuou-se um experimento no qual foram observados 20 erros de cada classe: os 10 erros que ocorrem com maior probabilidade, e os 10 erros que ocorrem com menor probabilidade. Ou seja, foram observados os 10 erros no topo e os 10 erros na base da região correspondente a cada classe, que podem ser identificadas na Figura 37. Outros 10 erros foram selecionados aleatoriamente, formando o

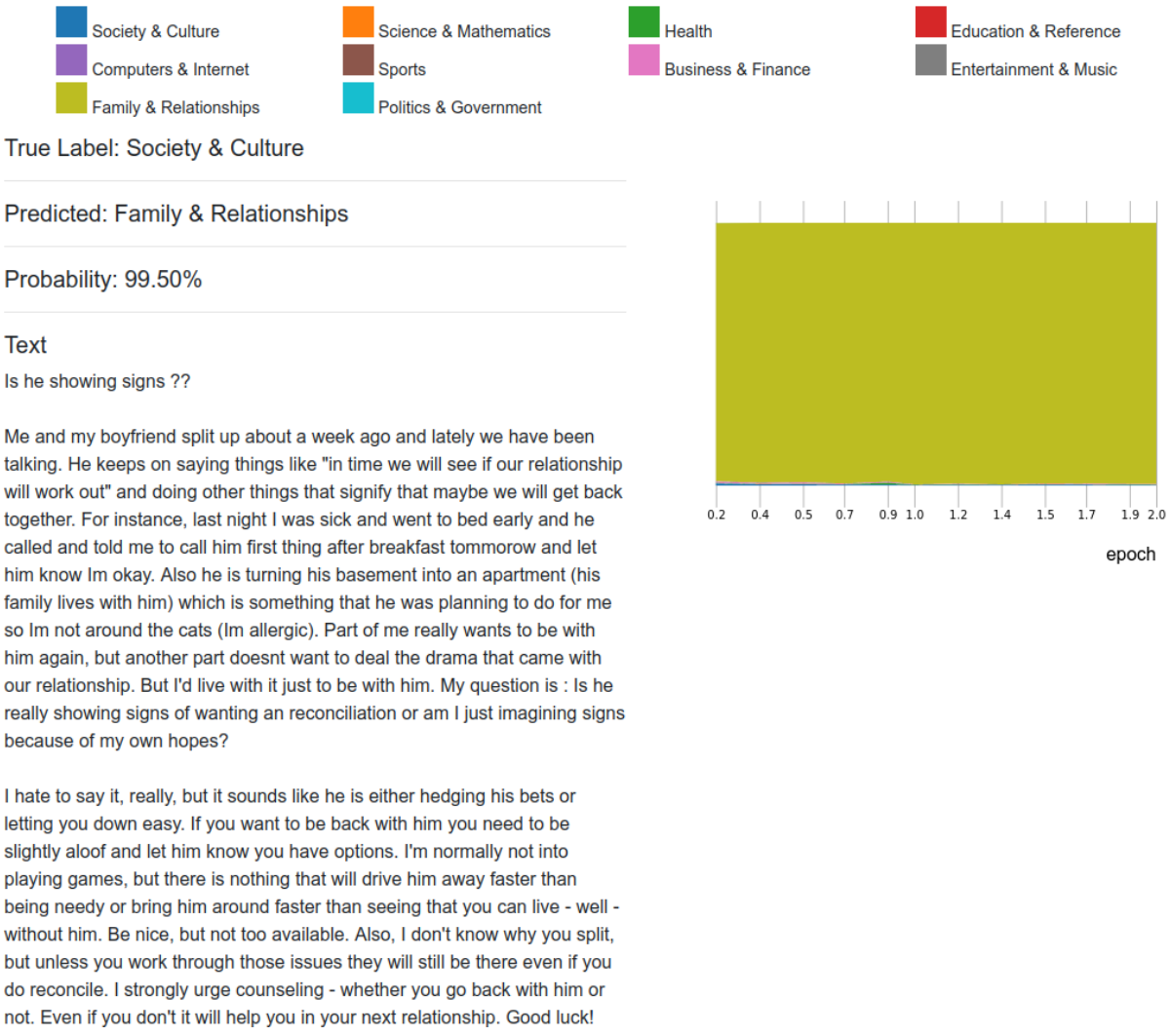


Figura 39 – Detalhes da instância mais ao topo e mais à esquerda, um caso mal rotulado

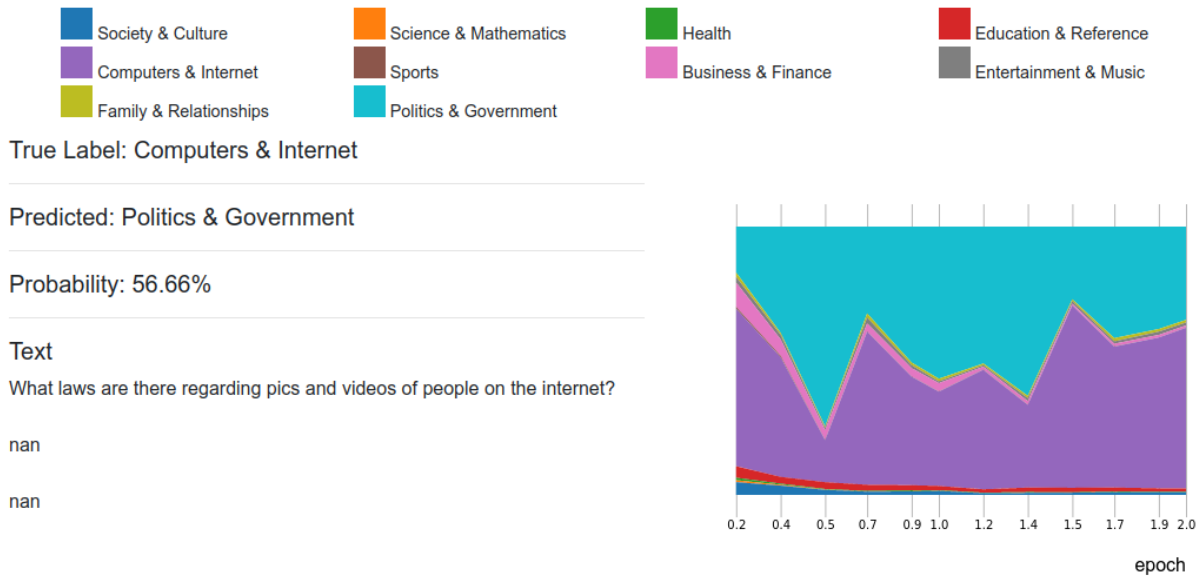


Figura 40 – Detalhes de uma instância ambígua, que poderia estar associada tanto à "Computers & Internet" quanto "Politics & Government". Este exemplo contém apenas o título da pergunta, sem conteúdo nem resposta

grupo de controle. No total, 300 casos foram observados e registrados. Cada um foi registrado como uma das opções a seguir.

- Mal rotulado: o rótulo está errado. Este tipo de observação também pode ser registrado como "Modelo acertou", caso a predição do modelo seja a classe correta de fato;
- Ambíguo: o texto deveria pertencer a mais de uma classe;
- Erros de fato: o rótulo está correto e verificou-se que trata-se de um erro verdadeiro do modelo;
- Textos ruins: são textos mal escritos, ou que não pertencem à nenhuma das classes do dataset.

A Tabela 2 mostra o resultado geral, somando todas as classes, discriminando entre os casos do topo, casos da base, e casos selecionados aletatoriamente. Nota-se que o dataset possui muitos erros de anotação, pois dos 100 casos observados na seleção aleatória, 41 estão mal rotulados. Porém a parte superior dos mapas mostrou-se extremamente eficaz na identificação de casos problemáticos, pois apenas 1 caso entre 100 é um erro de fato do modelo, com 80 sendo casos mal rotulados - em todos estes modelo indica o rótulo correto - e os outros 19 são textos ambíguos ou ruins. A parte inferior do mapa apresentou um bom potencial para a identificação de textos ruins, que poderiam ser removidos do dataset a fim de obtermos maior qualidade.

Tabela 2 – Somatório de todas as observações, discriminadas pela posição dos erros no mapa. Também apresenta os resultados acumulados das três classes com melhor desempenho e as três piores

Classe	Posição	Total	Mal rotulados	Modelo acertou	Ambíguos	Erros de fato	Textos ruins
Todas	Superior	100	80	80	18	1	1
	Inferior	100	29	10	14	28	28
	Aleatória	100	41	34	27	25	6
Três classes com melhor desempenho	Superior	30	19	19	10	1	0
	Inferior	30	8	3	2	6	14
	Aleatória	30	12	8	6	11	1
Três classes com pior desempenho	Superior	30	28	28	2	0	0
	Inferior	30	10	3	4	9	6
	Aleatória	30	13	13	9	4	3

Entre os resultados acumulados das classes com os melhores e os piores desempenhos, a única diferença observada é na capacidade de encontrar textos ruins: as classes com melhor desempenho apontam mais casos deste tipo na porção inferior do mapa. Contudo a amostra observada é pequena, e uma quantidade maior é necessária para tomar-se uma conclusão estatisticamente válida.

A Tabela 3 mostra o resultado separado por classe, também discriminados entre os casos do topo, casos da base, e casos selecionados aleatoriamente.

Tabela 3 – Contabilização das observações no mapa de instâncias: 10 erros do topo, 10 erros da base, e 10 erros aleatórios de cada classe

Classe	Posição	Total	Mal rotulados	Modelo acertou	Ambíguos	Erros de fato	Textos ruins
Society & Culture	Superior	10	8	8	2	0	0
	Inferior	10	2	1	1	5	2
	Aleatória	10	2	2	4	3	1
Science & Mathematics	Superior	10	10	10	0	0	0
	Inferior	10	3	1	3	2	2
	Aleatória	10	5	4	2	3	0
Health	Superior	10	7	7	2	0	1
	Inferior	10	2	1	4	4	0
	Aleatória	10	1	1	4	3	2
Education & Reference	Superior	10	10	10	0	0	0
	Inferior	10	4	1	0	4	2
	Aleatória	10	6	6	3	0	1
Computer & Internet	Superior	10	9	9	1	0	0
	Inferior	10	1	0	1	2	6
	Aleatória	10	2	1	3	5	0
Sports	Superior	10	4	4	5	1	0
	Inferior	10	2	1	1	1	6
	Aleatória	10	5	4	1	3	1
Business & Finance	Superior	10	10	10	0	0	0
	Inferior	10	4	1	3	0	2
	Aleatória	10	5	5	2	1	1
Entertainment & Music	Superior	10	8	8	2	0	0
	Inferior	10	3	1	1	3	3
	Aleatória	10	7	7	3	0	0
Family & Relationships	Superior	10	8	8	2	0	0
	Inferior	10	3	1	0	4	3
	Aleatória	10	3	1	3	4	0
Politics & Government	Superior	10	6	6	4	0	0
	Inferior	10	5	2	0	3	2
	Aleatória	10	5	3	2	3	0

A Figura 41 mostra a relação entre o desempenho da classe - precisão, recall e f1-score - e a quantidade de cada tipo de observação (eixo y). Dentro de cada gráfico, separa-se ainda de acordo com a posição no mapa da classe. Na primeira linha, correspondente às observações mal rotuladas, nota-se uma predominância de pontos azuis no topo dos gráficos, confirmando que a porção superior dos mapas é propícia para encontrar casos mal rotulados, mas ainda é possível observar que o desempenho da classe não importa. Tanto faz se o f1-score está na casa dos 85% ou por volta de 70%. As observações nas quais o modelo acertou seguem um padrão parecido.

Figura 41 – Relação entre o desempenho de cada classe e a quantidade de cada tipo de observação registrada, separadas de acordo com a posição no mapa da classe

5 Identificação de Vieses e Qualidade do Dataset

Este capítulo discute a aplicação da técnica de saliência Integrated Gradients (SUNDARARAJAN; TALY; YAN, 2017), apresentada na Seção 3.1.2, para a análise dos erros do classificador observados no experimento do uso do mapa de instâncias como revisor de anotações, descrito na Seção 4.3. O objetivo principal é investigar se a técnica de saliência empregada ajuda na identificação de vieses não-intencionais incorporados pelo modelo durante o treinamento. Para esta finalidade realizou-se a observação dos erros no topo de cada classe, comparando as saliências obtidas para o modelo treinado por uma época com as saliências do modelo treinado por duas épocas. Uma maior ênfase no modelo treinado por duas épocas pode indicar um viés presente no dataset de treinamento.

Como objetivos secundários é interessante a observação dos erros que ocorrem na base dos mapas, onde o modelo não conseguiu identificar características próprias de nenhuma classe e portanto está indeciso entre várias delas, assim como os casos em que o classificador está com probabilidade próxima de 50% para duas classes, ou seja, onde está dividido entre apenas duas. A comparação das saliênicias para cada classe pode revelar qual parte do texto o modelo considera como característico de cada classe.

Após a identificação dos vieses, o dataset de treinamento é refeito para balancear a presença dos termos identificados. No entanto, diferentemente do trabalho de (DIXON et al., 2018), no qual adicionam-se novos textos de uma classe diferente, no presente trabalho foram removidos textos que contém a palavra enviesada até obtermos a mesma proporção no dataset completo e na classe para a qual o viés foi identificado. Por exemplo, o termo "health" aparece em 15 mil textos da classe "Health" e em 27 mil textos das outras nove classes juntas. No entanto trata-se de um termo genérico, e sua simples aparição no texto não deve ser indicativo de tratar-se da classe "Health". O novo dataset de treinamento é feito removendo-se 12 mil textos com a palavra "health" dentre os rotulados como "Health", sobrando assim 3 mil, que é a mesma proporção por classe — em média — no restante do dataset. A avaliação dos resultados é feita de forma qualitativa pelo Integrated Gradients, e de forma quantitativa pelas taxas de falsos positivos e falsos negativos associadas a cada par termo-classe.

Como apresentado nesta seção, a visualização por Integrated Gradients mostrou-se capaz de identificar as características que o BERT utiliza para fazer as sua predições, auxiliando na interpretação de suas decisões. Isso possibilitou a observação das palavras que causam o overfitting do modelo. A mitigação dos vieses mostrou-se eficaz para a diminuição dos falsos positivos associados às palavras enviesadas, porém ao custo de um aumento na taxa de falsos negativos.

Esta Seção segue com a apresentação da metodologia da análise realizada, com o detalhamento da implementação do Integrated Gradients. Segue, então, o registro e discussão do todos os casos observados, com uma analíse quantitativa que apoia as observações, e por fim tem-se o resultado da mitigação dos vieses.

5.1 Metodologia da análise

A análise pode ser separada em três tipos distintos de erros: i) erros no topo dos mapas de instâncias; ii) erros na base dos mapas; iii) erros quase-acertos. Os erros no topo dos mapas possuem a finalidade de identificação dos viéses não-intencionais, mas assim como os demais tipos de erros, são interessantes também para entendermos o motivo pelo qual o modelo escolhe cada classe. Estes erros possuem uma probabilidade muito alta para a classe predita, tratando-se de textos característicos da classe aos olhos do modelo. Portanto, são textos interessantes para a identificação de vieses presentes no conjunto de treinamento. Além disso, a aplicação do Integrated Gradients para a comparação entre o modelo treinado por uma época e o modelo treinado por duas épocas pode apontar os vieses, pois como visto no Apêndice A, o modelo sofre um overfitting extremo na segunda época de treinamento.

Já no caso dos erros da base, e dos erros quase-acertos, é interessante comparar as saliências da classe predita e do rótulo, a fim de entender a confusão do classificador. A comparação entre os modelos treinados por uma e duas épocas é realizada quando o streamgraph aponta diferenças significativas entre as probabilidades na época 1 e na época 2.

De acordo com a categorização geral discutida por (DANILEVSKY et al., 2020), apresentada na Seção 2.2.3.2, a análise aplicada é local, pois trata-se de uma avaliação de casos específicos do dataset, e post-hoc, pois necessita do cálculo da derivada por backpropagation após o processo de predição. E de acordo com as categorias específicas para NLP propostas também por (DANILEVSKY et al., 2020), e apresentadas na Seção 2.2.3.3, a técnica de explicabilidade dá-se pela importância de características; a operação de explicabilidade é a saliência da primeira derivada; e a visualização é de saliência.

5.2 Implementação do Integrated Gradients

Nesta seção descreve-se a implementação do Integrated Gradients que gerou as observações deste trabalho.

O vetor baseline utilizado, para o qual teoricamente não há preferência na classificação para nenhuma classe, ou seja, um embedding nulo, é um vetor de zeros: um array unidimensional com 768 posições, que é o tamanho do embedding do BERT base, somente com números zero.

A derivada considerada em cada token é referente à soma do embedding do token com seu embedding posicional, e ainda com o embedding do segmento, ou seja, o resultado da

soma mostrada na Figura 17 da Seção 2.2.2.5. Assim, um mesmo token apresenta importâncias diferentes conforme a sua posição na sentença. Para obtê-la, efetua-se uma inferência com os embeddings de entrada, e então toma-se a derivada em relação ao logit da classe de interesse por backpropagation. Dessa forma, cada classe apresenta uma saliência diferente sobre o texto, mostrando a parte mais relevante para cada uma.

Para fazer a soma dos embeddings parciais, considerou-se 10 iterações, conforme descrito na Seção 3.1.2. Cada saliência resulta da soma de dez derivadas, variando-se linearmente de 0 a 1 - ou seja, com incrementos de 0,1 - o peso que multiplica os embeddings da entrada.

Para visualizar a saliência em cada caso, toma-se o valor mínimo e o valor máximo obtido para todos os tokens da sentença, convertendo os valores para uma escala de cinza entre 0 e 255. Dessa forma, termos relevantes aparecem em preto, e termos pouco importantes ficam apagados contra o fundo branco.

5.3 Observação dos erros

Nesta seção estão registradas as observações dos erros realizadas com o Integrated Gradients, separadas de acordo com a posição no mapa. Nos casos do topo do mapa, onde apenas a classe predita possui alta probabilidade, apenas a saliência para a classe predita é analisada, comparando-a no modelo treinado por uma e por duas épocas. Já para os erros da base, e para os erros quase-acertos, também observa-se outras classes que não a predita, mas que apresentam probabilidades relevantes.

A Figura 42 mostra a região da classe "Health" no Mapa de Instâncias. Os erros do topo são encontrados ao percorrer o mapa da esquerda para a direita e de cima para baixo. Nota-se que a primeira instância já não foi classificado como sendo da classe "Health". Os erros da base são encontrados no sentido contrário, ou seja, da direita para a esquerda e de baixo para cima. A grande maioria foi classificada como sendo de outras classes. E os erros quase-acertos são as instâncias coloridas com outras cores que não a verde, para as quais a probabilidade da predição esteja próxima de 50% para duas classes.

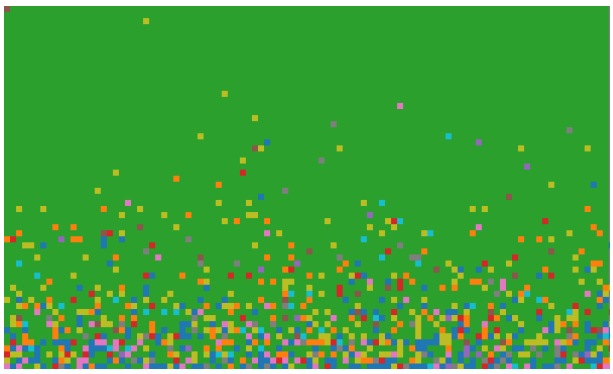


Figura 42 – Região da classe "Health" do Mapa de Instâncias

Além dos modelos treinados com o dataset padrão, também visualizam-se as saliências para os modelos treinados com o dataset preparado para a mitigação dos vieses. Na maioria dos casos nota-se como a saliência ficou mais distribuída após a mitigação dos vieses, indicando que o modelo apoia-se em mais características para tomar suas decisões; um indicativo de que ele tornou-se mais robusto.

5.3.1 Erros no topo

O primeiro caso de overfitting observado é a palavra "back", que torna-se muito mais saliente ao prolongar o treinamento por mais uma época. A Figura 43a mostra o exemplo observado, conforme exibido no Mapa de Instâncias. A probabilidade para a classe "Health" - que deveria ser o rótulo ao invés de "Sports" - é bastante alta. As Figuras 43b e 43c mostram a saliência para a classificação da classe "Health" no modelo treinado por uma época, e também por duas épocas. Observa-se que o modelo treinado por mais tempo foca apenas no termo "back", enquanto com apenas uma época de treinamento a saliência é muito mais distribuída, embora concentrada também em "back".

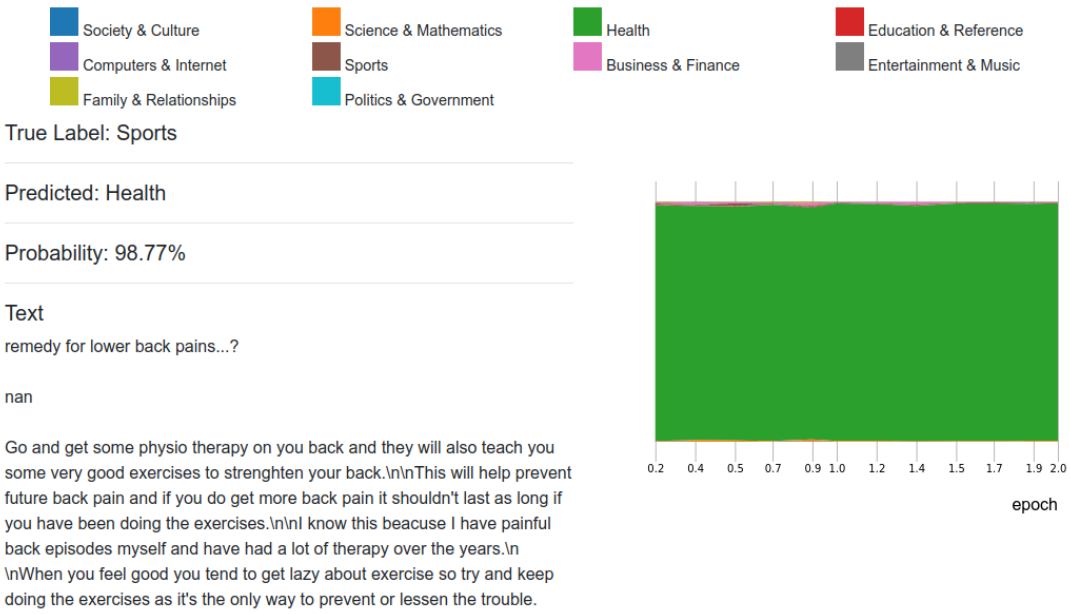
A palavra "back", no sentido de costas ou tronco humano, não é exclusividade da classe "Health", pois também é característica do tema "Sports". Dessa forma, observa-se aqui um viés não-intencional aprendido pelo modelo. Como apresentado na Tabela 4, de fato a palavra "back" aparece em 10% dos exemplos da classe, contra 7% na média do dataset inteiro.

As Figuras 44a e 44b mostram a saliência do modelo treinado após a identificação e mitigação dos vieses. Ambos os casos continuam sendo classificados como "Health", com probabilidade de 97%, no entanto é notável que ele se apoia em muito mais palavras para tomar a sua decisão.

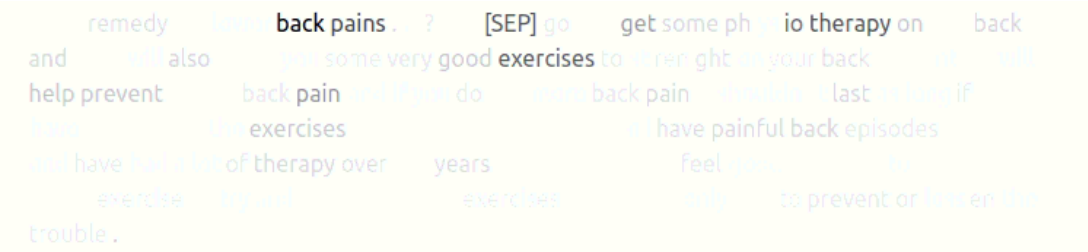
A palavra "back" tornou-se muito menos importante, nem estando entre as mais relevantes. E a saliência parece bem mais sólida: "exercise", "physiotherapy", "therapy" e "painful" são palavras importantes para "Health" que não eram vistas antes. Ou seja, de acordo com a saliência, o viés não-intencional foi mitigado, tornando o modelo mais robusto.

Outro forte viés observado é a palavra "thigh", que também está puxando a classificação para "Health". Trata-se de outro viés não-intencional, pois "thigh" também pode estar relacionado com "Sports". A Figura 45 mostra o texto e as saliências da classificação para os modelos dos dois checkpoints.

Após a tentativa de mitigar os vieses, a saliência do modelo para a primeira época mudou pouco. Ou seja, a equalização das ocorrências dos tokens por classe nem sempre resulta em uma distribuição da saliência. Já a saliência do modelo treinado por duas épocas ficou concentrada no token separador "[SEP]", como mostra a Figura 46. A predição continuou sendo "Health", com 94,53% de probabilidade no modelo treinado por uma época, e 97,49% no que foi treinado por duas épocas.



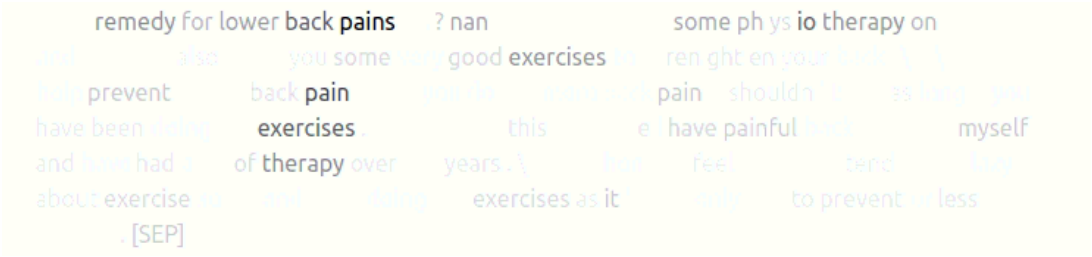
(a) Detalhes do exemplo



(b) Saliência para classificação de "Health" com o modelo treinado por 1 época

(c) Saliência para classificação de "Health" com o modelo treinado por 2 épocas

Figura 43 – Exemplo de overfitting relativo à palavra "back" para classificação da classe "Health"



(a) Saliência para classificação de “Health” com o modelo treinado por 1 época

(b) Saliência para classificação de “Health” com o modelo treinado por 2 épocas

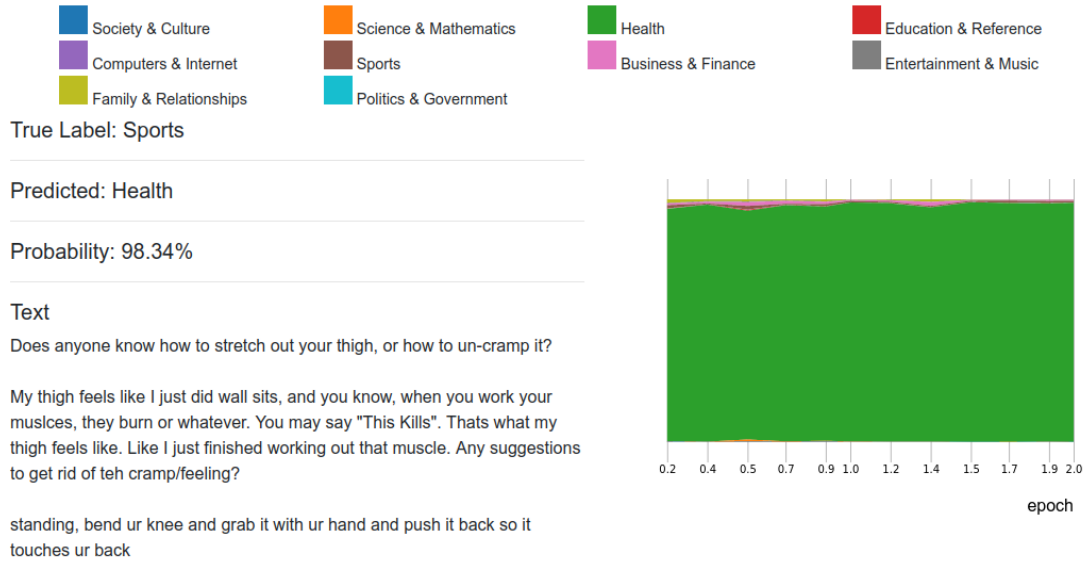
Figura 44 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra “back”.

Um terceiro erro do topo dos casos rotulados como “Sports” apresenta alta saliência para o termo “medicine”, classificando-o como “Health”. Este é um viés intencional, pois o termo “medicine” não é altamente relevante para nenhuma das outras classes. A Figura 47 mostra o exemplo, como exibido no Mapa de Instâncias, e as saliências para as predições dos modelos de uma e duas épocas. Nota-se que “medicine” está em grande destaque, com aparições muito discretas das outras palavras.

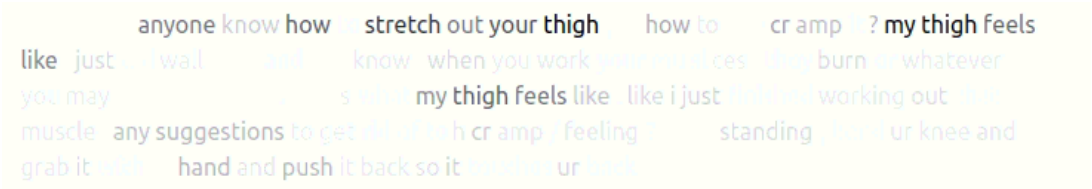
Após retreinar o BERT com o dataset para mitigação dos vieses, a saliência para a palavra “medicine” reduziu muito para o modelo de uma época, que passou a destacar palavras como “headache” e “pill”, como mostra a Figura 48 indicando que o modelo generalizou melhor. O modelo de duas épocas continua com saliência maior relacionada à “medicine”, com exceção da parte principal, que é o token separador. No entanto a probabilidade da predição caiu para 74,50% no modelo de uma época, e 48,05% no de duas épocas.

A Figura 49 mostra um caso rotulado como “Education & Reference”, mas que claramente trata-se de “Computers & Internet”, classe corretamente dada pelo BERT treinado. A saliência destaca bastante a palavra “data”, embora outras palavras relacionadas ao tema, como “files”, “cookies”, “download”, “settings” e computer code” também apareçam.

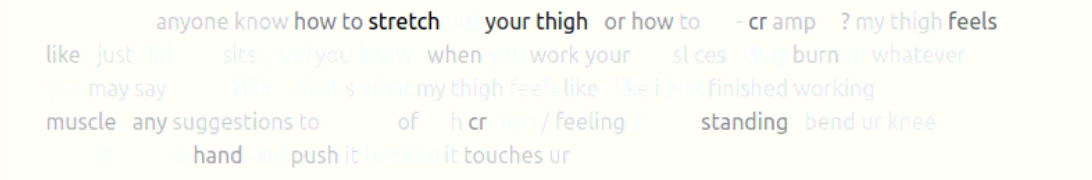
A palavra “data” de fato é bem característica da classe “Computers & Internet”, porém é muito genérica e pode ser usada em vários contextos nas demais classes. Por isso foi considerada como um viés não-intencional. A Figura 50 mostra as saliências para os modelos treinados a fim de mitigar o viés: o termo “data” fica sem destaque algum. Várias outras palavras relevantes aparecem com mais destaque no modelo de uma época, indicando que o modelo distribuiu mais a sua decisão.



(a) Detalhes do exemplo

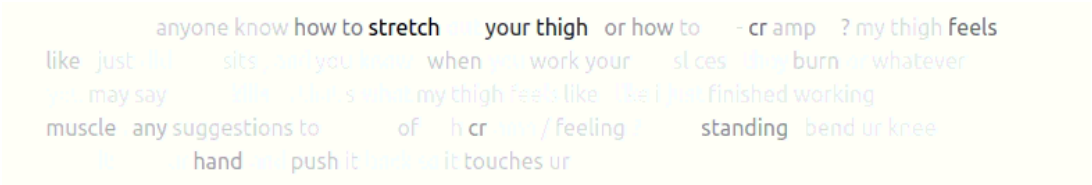


(b) Saliência para classificação de “Health” com o modelo treinado por 1 época



(c) Saliência para classificação de “Health” com o modelo treinado por 2 épocas

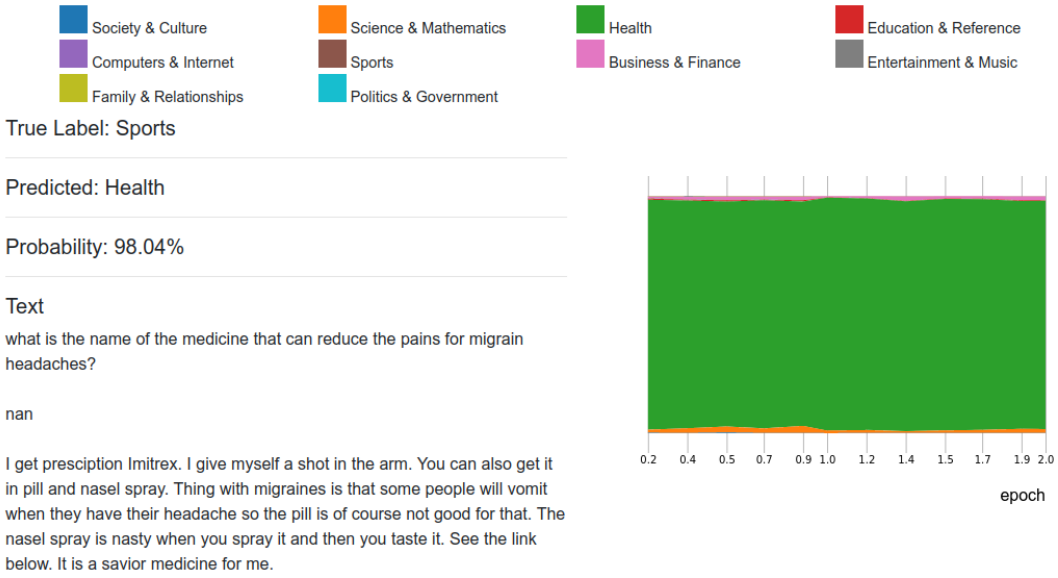
Figura 45 – Exemplo de overfitting relativo à palavra “thigh” para classificação da classe “Health”



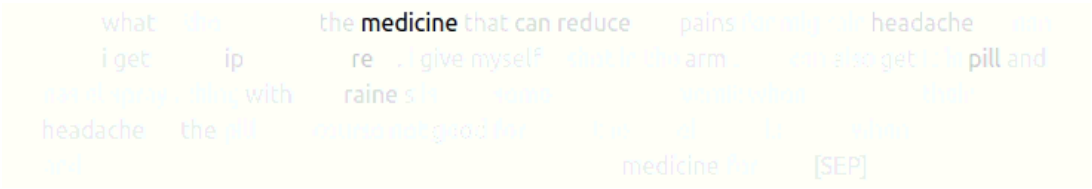
(a) Saliência para classificação de “Health” com o modelo treinado por 1 época

(b) Saliência para classificação de “Health” com o modelo treinado por 2 épocas

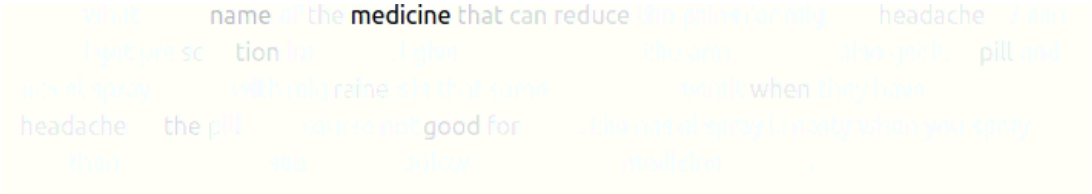
Figura 46 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra “thigh”.



(a) Detalhes do exemplo

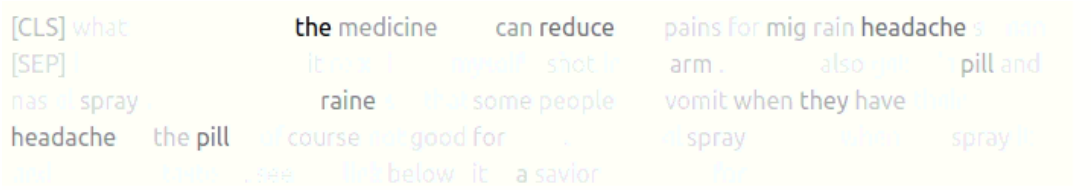


(b) Saliência para classificação de "Health" com o modelo treinado por 1 época

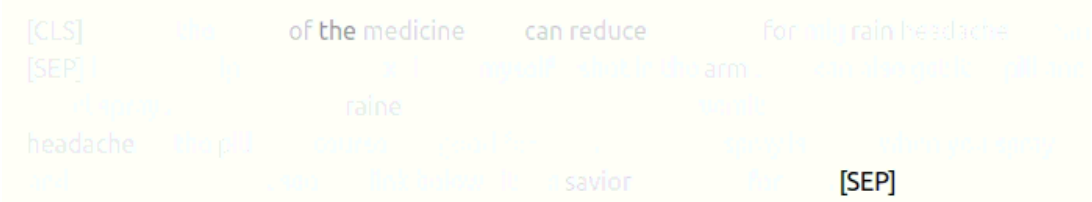


(c) Saliência para classificação de "Health" com o modelo treinado por 2 épocas

Figura 47 – Exemplo de overfitting relativo à palavra "medicine" para classificação da classe "Health"

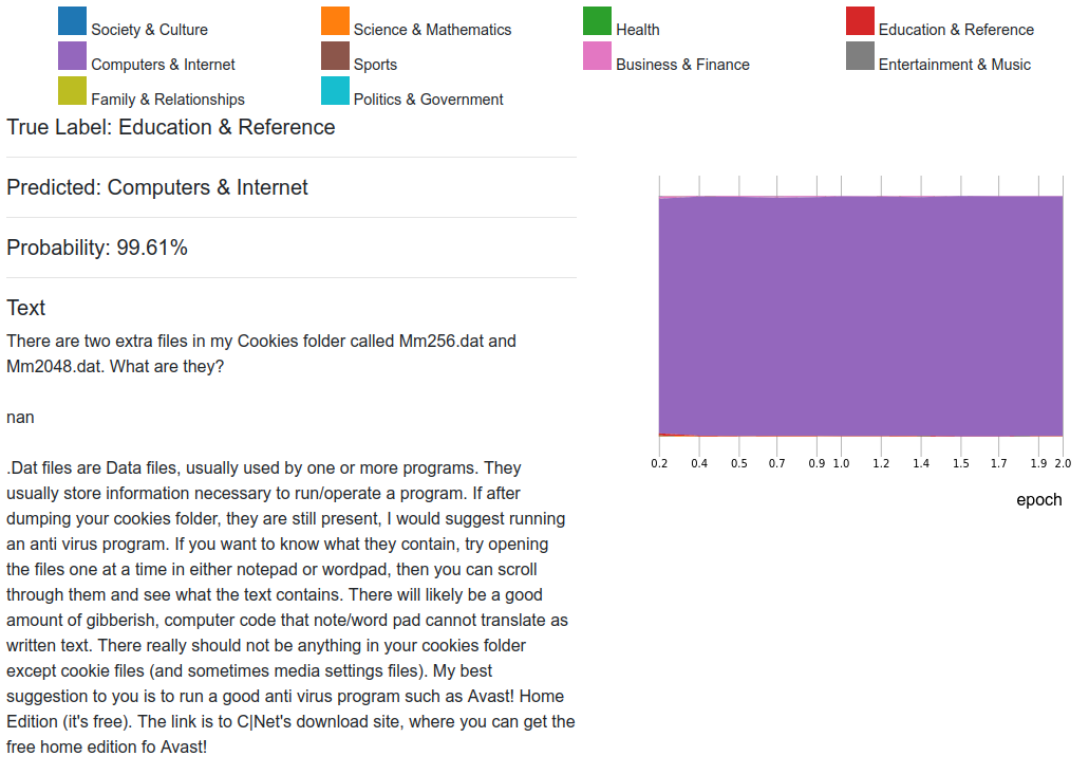


(a) Saliência para classificação de "Health" com o modelo treinado por 1 época



(b) Saliência para classificação de "Health" com o modelo treinado por 2 épocas

Figura 48 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra "medicine".



(a) Detalhes do exemplo

(b) Saliência para classificação de "Computers & Internet" com o modelo treinado por 1 época

(c) Saliência para classificação de "Computers & Internet" com o modelo treinado por 2 épocas

Figura 49 – Exemplo de overfitting relativo à palavra "data" para classificação da classe "Computers & Internet"

(a) Saliência para classificação de “Computers & Internet” com o modelo treinado por 1 época

(b) Saliência para classificação de “Computers & Internet” com o modelo treinado por 2 épocas

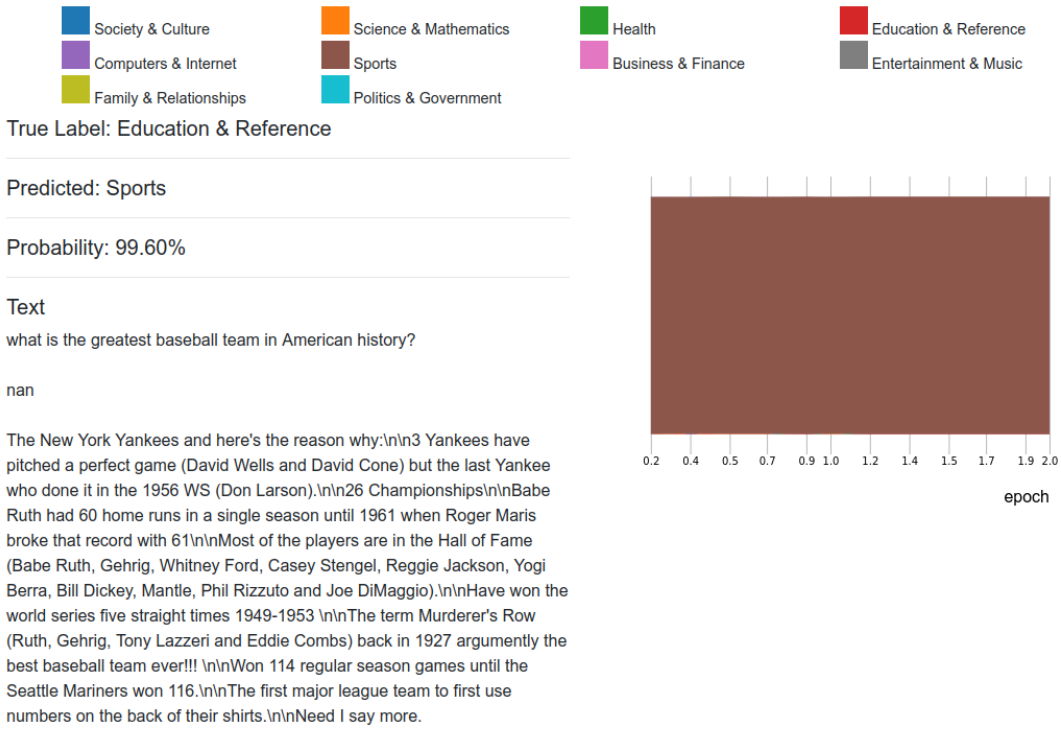
Figura 50 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra “data”.

A Figura 51 mostra um caso onde a palavra “baseball” é praticamente exclusiva para a decisão de classificar como “Sports”. Não há muita diferença entre o modelo de uma época e o de duas.

Apesar de tratar-se de um viés intencional, o efeito da mitigação também foi incluído no experimento. A Figura 52 mostra as saliências do novo BERT. O termo “baseball” continua em evidencia quase exclusiva. Ou seja, mesmo equilibrando a quantidade de ocorrências em relação ao restante do dataset, o modelo continua com o mesmo foco na palavra.

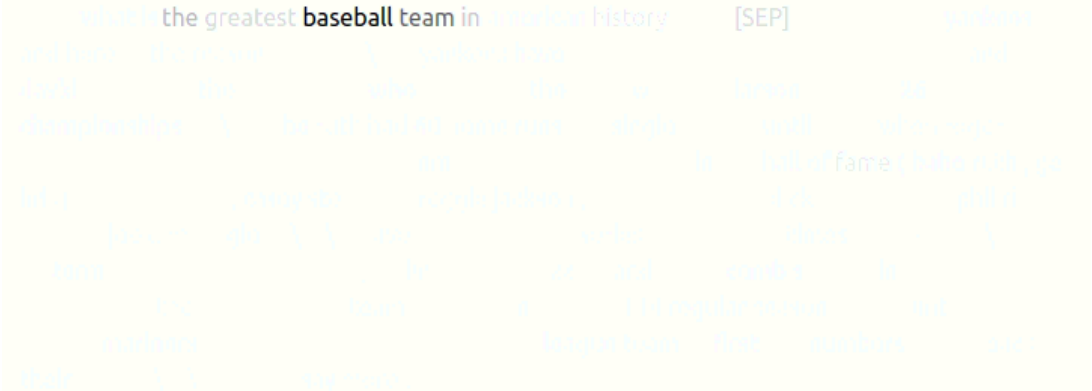
Outro caso interessante é o termo “802”, parte do nome do protocolo de internet “802.11b”, que aparece no texto da Figura 53. Trata-se de um caso rotulado como “Education & Reference”, mas que na verdade é da classe “Computers & Internet”. A saliência aponta uma importância muito grande para o termo “802”, quando existem muitos outros elementos no texto que poderiam ter levado à classificação como “Computers & Internet”, como “connection”, que aparece levemente saliente, e “internet access”, que não aparece na saliência.

Após a mitigação dos vieses, nota-se que o termo “802” deixou de ser tão importante, com as palavras “connection”, e “internet access” ganhando maior relevância, como mostra a



(a) Detalhes do exemplo

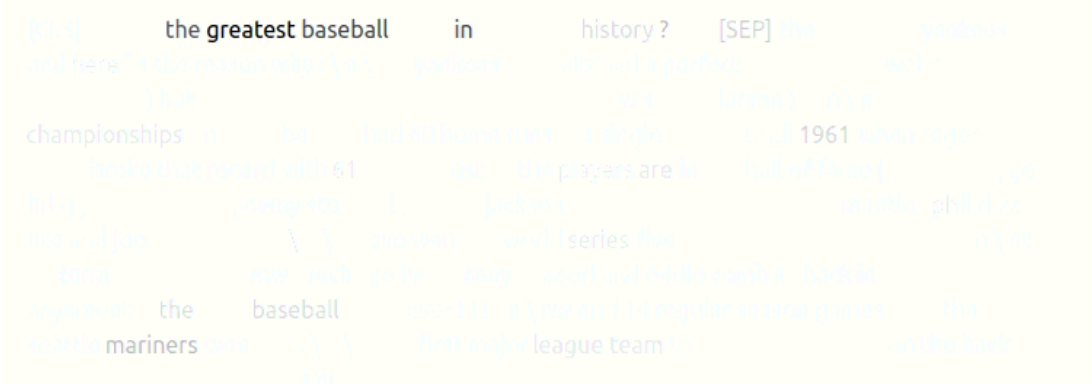
(b) Saliência para classificação de “Sports” com o modelo treinado por 1 época



(c) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas

Figura 51 – Exemplo de overfitting relativo à palavra “baseball” para classificação da classe “Sports”

(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época



(b) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas

Figura 52 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra “baseball”.

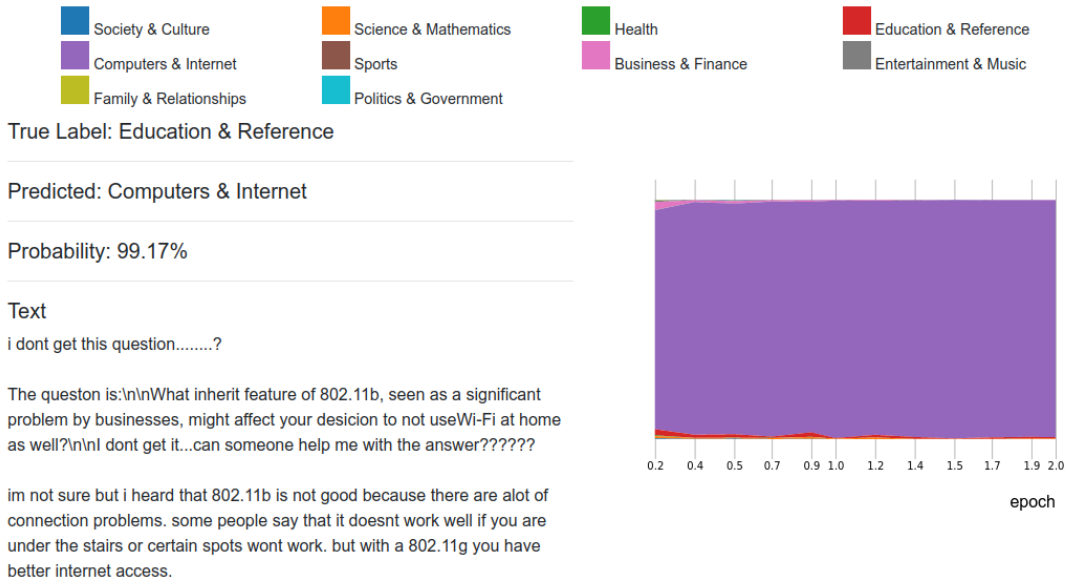
Figura 54.

5.3.2 Erros na base

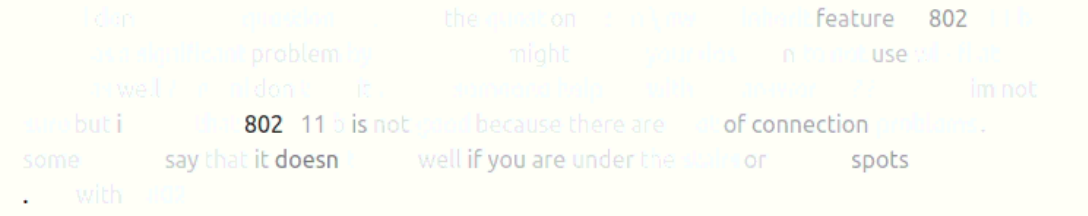
O perfil dos erros da base é bem diferente dos erros do topo, pois a probabilidade da classificação está muito baixa, e muitas vezes próxima da probabilidade de outras classes. As saliências não são interpretáveis na maioria dos casos, e por isso apenas dois casos são discutidos nesta seção. O fato delas não serem interpretáveis pode estar relacionado com o fato de estarem na base dos mapas pelo motivo de nenhuma característica significativa ter sido encontrada para nenhuma classe.

A Figura 55 mostra um caso interessante em que o modelo está bem indeciso com uma época de treinamento, mas durante a segunda época tende a classificar fortemente como “Sports”.A Figura 56 exibe as saliências visualizadas, onde nota-se que o termo mais importante para esta classificação é “teams”. Após a mitigação dos vieses, “teams” continua uma palavra importante para “Sports”, porém a saliência está muito mais distribuída.

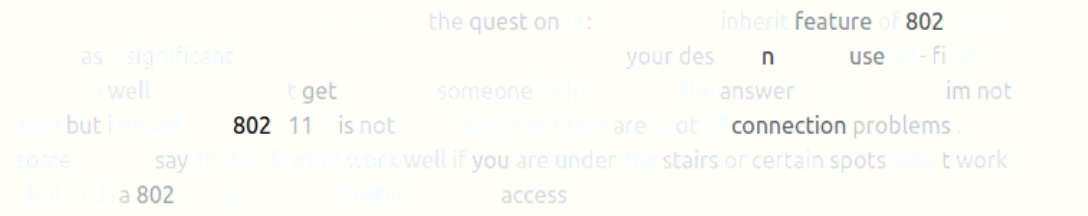
O streamgraph da Figura 57 mostra a distribuição de probabilidades ao longo do treinamento do modelo após a mitigação dos vieses. Nota-se como a probabilidade de “Sports” caiu para praticamente zero, deixando o modelo indeciso entre “Business & Finance” e “Science



(a) Detalhes do exemplo

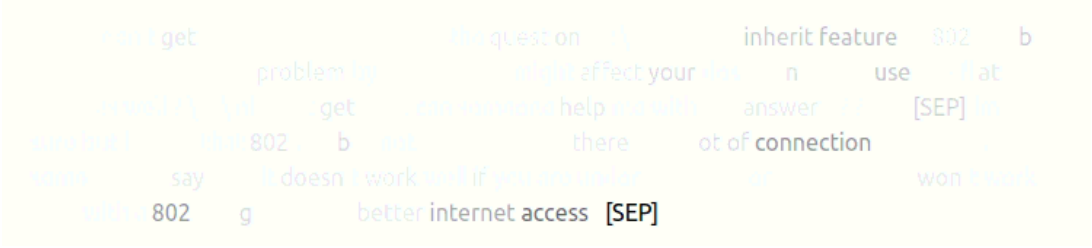


(b) Saliência para classificação de "Computers & Internet" com o modelo treinado por 1 época

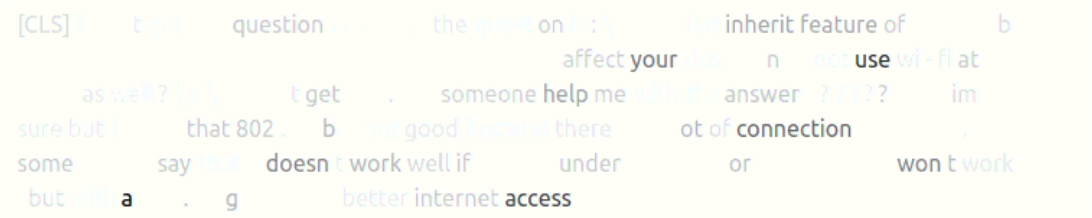


(c) Saliência para classificação de "Computers & Internet" com o modelo treinado por 2 épocas

Figura 53 – Exemplo de overfitting relativo à palavra "802" para classificação da classe "Computers & Internet"



(a) Saliência para classificação de “Computers & Internet” com o modelo treinado por 1 época



(b) Saliência para classificação de “Computers & Internet” com o modelo treinado por 2 épocas

Figura 54 – Saliência após a mitigação dos vieses. Antes era muito concentrada na palavra “802”.

& Mathematics” na primeira época.

O segundo caso da base dos mapas que mostra um forte viés é o termo “community” sendo usado para classificar o texto como “Education & Reference”. A Figura 58 mostra o texto e as saliências. No modelo de uma época a saliência saiu totalmente concentrada no token separador, tornando difícil sua interpretação, mas na segunda época vemos o forte viés. Após a mitigação dos vieses, nota-se que outras palavras relevantes tornaram-se importantes, como “scouts” e “camps”.

A Figura 59 mostra o streamgraph do modelo após a mitigação dos vieses. “ Education & Reference” passa a dar um pouco mais de espaço para “Sports”, mas não mudou muita coisa.

5.3.3 Erros quase-acertos

Para os erros quase-acertos é interessante observar a saliência para as duas classes mais prováveis, pois são casos onde o modelo praticamente empata entre elas. Conhecendo melhor a potencial causa da confusão, podemos tentar melhorar o classificador.

A Figura 60 mostra os detalhes de uma instância rotulada como “Sports”, mas que o modelo confunde com “Entertainment & Music” durante a primeira época de treinamento. O texto fala sobre a causa da morte de Eddie Guerrero, um lutador profissional. Portanto, tem elementos de várias classes: “Sports”, “Health” e “Entertainment & Music”.

As Figuras 61a e 61b mostram as saliências para as classificações do modelo treinado por uma época, para as classes “Sports” e “Entertainment & Music”, respectivamente. Nota-se

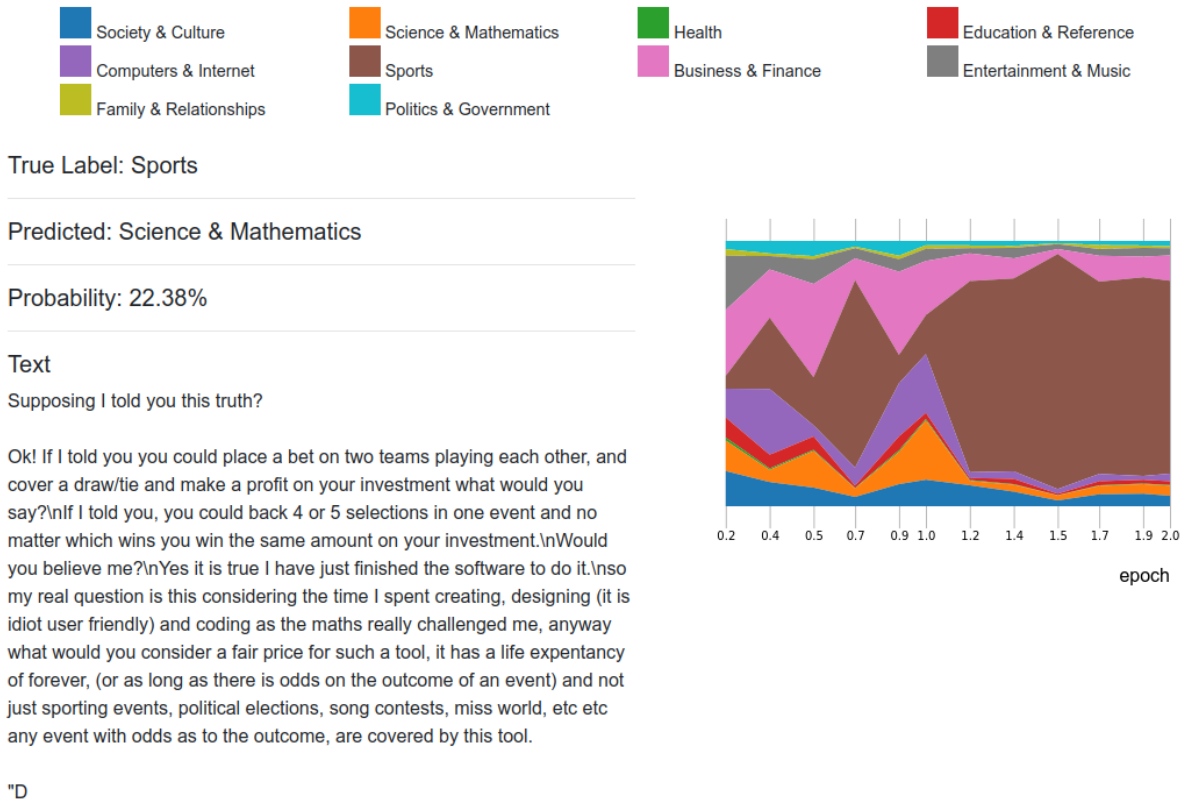


Figura 55 – Detalhes de uma instância dentre os erros da base

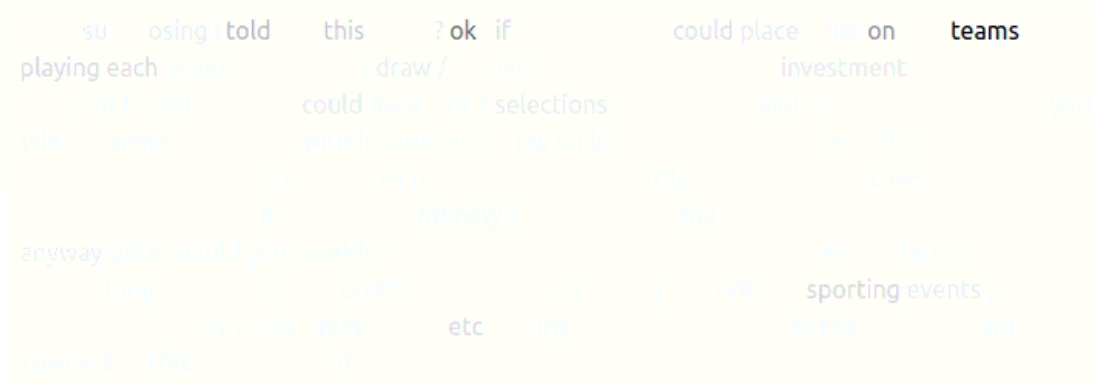
uma forte saliência para o nome "Eddie" na classe sobre entretenimento, juntamente com os termos "death" e "died" em menor relevância. Para "Sports" a saliência está bem distribuída.

Já as Figuras 62a e 62b mostram as saliências do modelo treinado por duas épocas. Para a classificação de "Sports", o token separador torna-se menos importante, enquanto para "Entertainment & Music" o foco principal continua no nome "Eddie".

Nota-se, portanto, que o nome "Eddie" é um viés que o modelo tem para classificar como "Entertainment & Music". Porém trata-se de um viés não-intencional, uma vez que nomes próprios não são características exclusivas desta classe. A Figura 63 mostra as saliências para o modelo treinado com o dataset para mitigação dos vieses. A classificação tornou-se "Sports", com 72,72% de probabilidade. Observa-se que "Eddie" acabou tornando-se saliente na classe "Sports", mas as saliências do modelo treinado por duas épocas ficaram totalmente no token separador, dificultando sua interpretação.

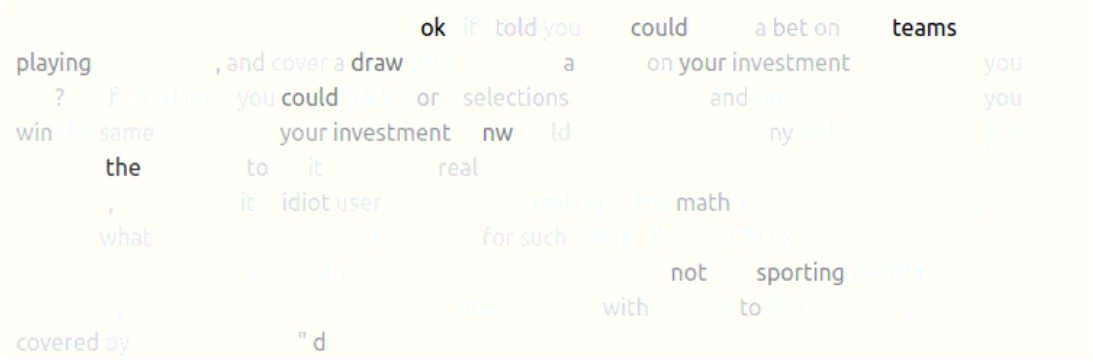
O streamgraph para este exemplo, após a mitigação dos vieses, mostra que a classe "Entertainment & Music" perdeu totalmente o espaço ao longo do treinamento, como mostra a Figura 64. O viés para classificar como "Sports" tornou-se muito mais forte.

Outra observação feita nos erros quase-acertos envolve também o termo "baseball" para a classe "Sports", como visto anteriormente no caso da Figura 51. O caso da Figura 65 fala de uma pesquisa sobre beisebol durante a Grande Depressão, na qual o modelo fica indeciso entre classificar como "Sports" ou "Education & Reference" na primeira época de treinamento.



(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época

(b) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas



(c) Saliência para classificação de “Sports” com o modelo treinado por 1 época, após a mitigação dos vieses

(d) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 56 – Viés de “teams” para classificar como “Sports”

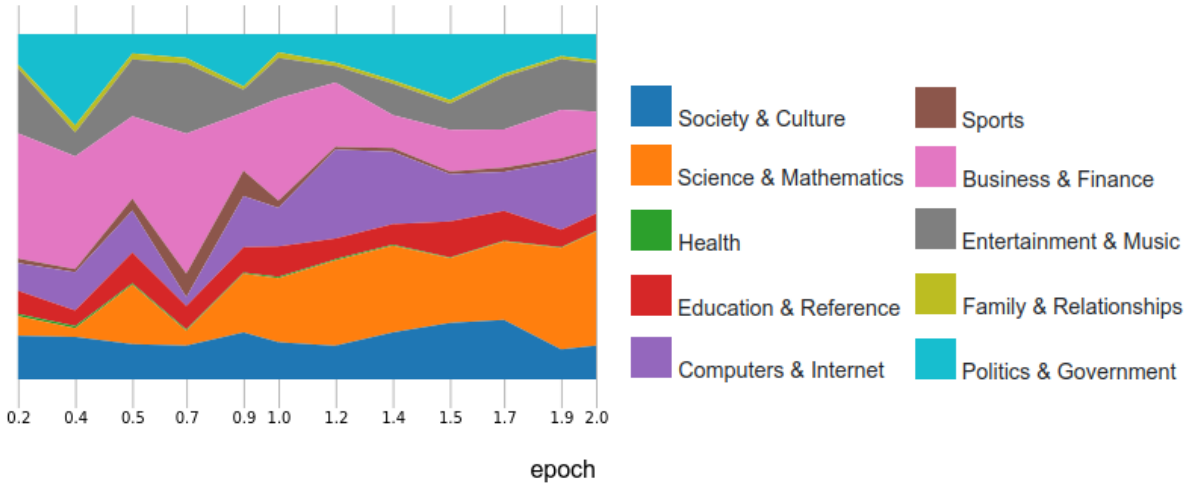


Figura 57 – Streamgraph, após a mitigação dos vieses, do caso onde o termo "teams" era muito característico da classe "Sports"

Após o overfitting, "Sports" torna-se a predição dominante, como mostra o streamgraph.

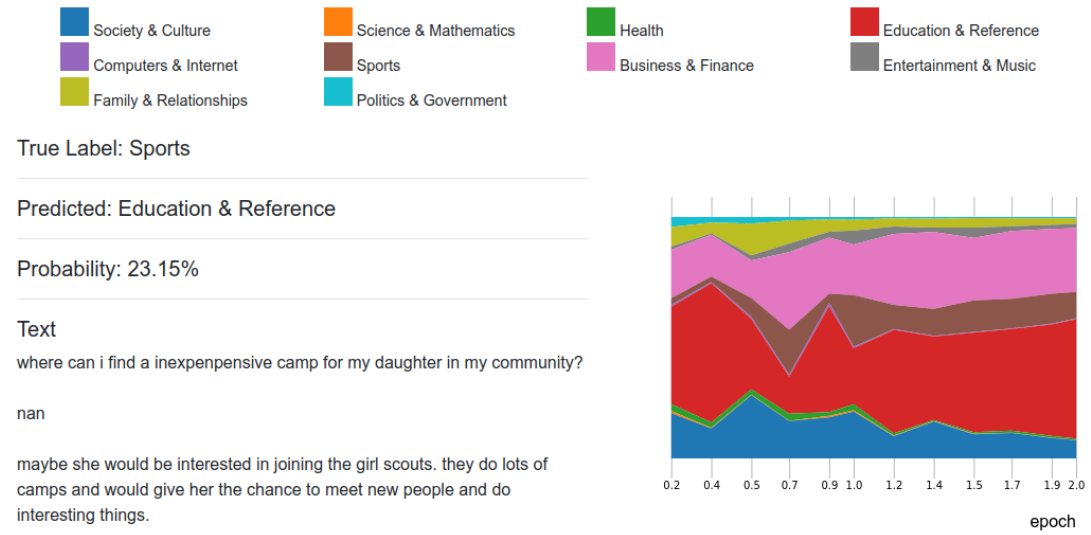
A Figura 66 compara as saliências para as classes "Sports" e "Education & Reference", para o modelo treinado por uma e por duas épocas. A classificação como "Sports" dá-se principalmente pelo termo "baseball", com pouco espaço para outras palavras. A razão de classificar como "Education & Reference" concentra-se no termo "great depression", mas distribui-se entre outras palavras. Comparando o modelo da primeira época com o da segunda, observa-se uma saliência ainda mais distribuída, indicando que para este caso específico o modelo treinado por duas épocas continua bem generalizado.

Assim como no outro caso do termo "baseball", a saliência nesta palavra continua forte após a mitigação dos vieses, como mostra a Figura 67. É um viés intencional, que manteve-se mesmo após o equilíbrio deste termo na classe, comparado à média do dataset.

A Figura 68 mostra o streamgraph após a mitigação dos vieses. "Education & Reference" passa a dominar as probabilidades de classificação.

Outro caso com alta saliência para um termo específico ocorre com o termo "health" e a classe "Health", observado para o exemplo exibido na Figura 69. O modelo fica entre "Health" e "Education & Reference" na sua classificação, como apresenta o streamgraph. O texto fala sobre um trabalho de escola em relação à saúde, sendo portanto ambíguo. Dessa forma, o modelo está correto na sua indecisão. Tornando-se interessante observar quais partes do texto estão associadas a cada classe.

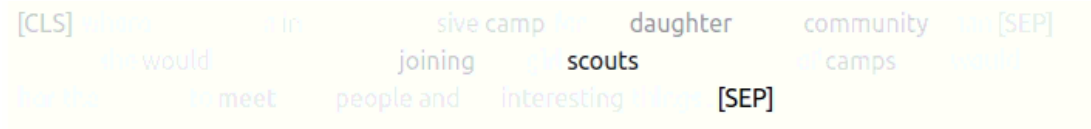
A Figura 70 mostra as saliências das classificações. Nota-se como "health" está muito mais destacada que as demais palavras para as classificações como "Health". Já "Education & Reference" apresenta uma saliência um pouco mais distribuída. Mas é interessante ver que cada classe apresenta saliência nos termos que fazem sentido.



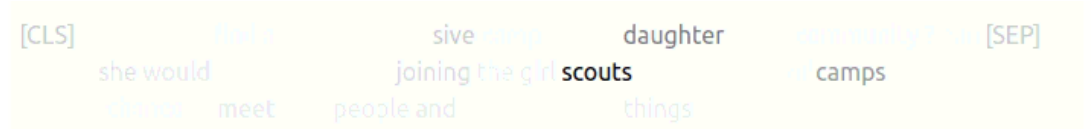
(a) Detalhes de uma instância

(b) Saliência para classificação de "Education & Reference" com o modelo treinado por 1 época

(c) Saliência para classificação de "Education & Reference" com o modelo treinado por 2 épocas



(d) Saliência para classificação de "Education & Reference" com o modelo treinado por 1 época, após a mitigação dos vieses



(e) Saliência para classificação de "Education & Reference" com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 58 – Viés de "community" para classificar como "Education & Reference"

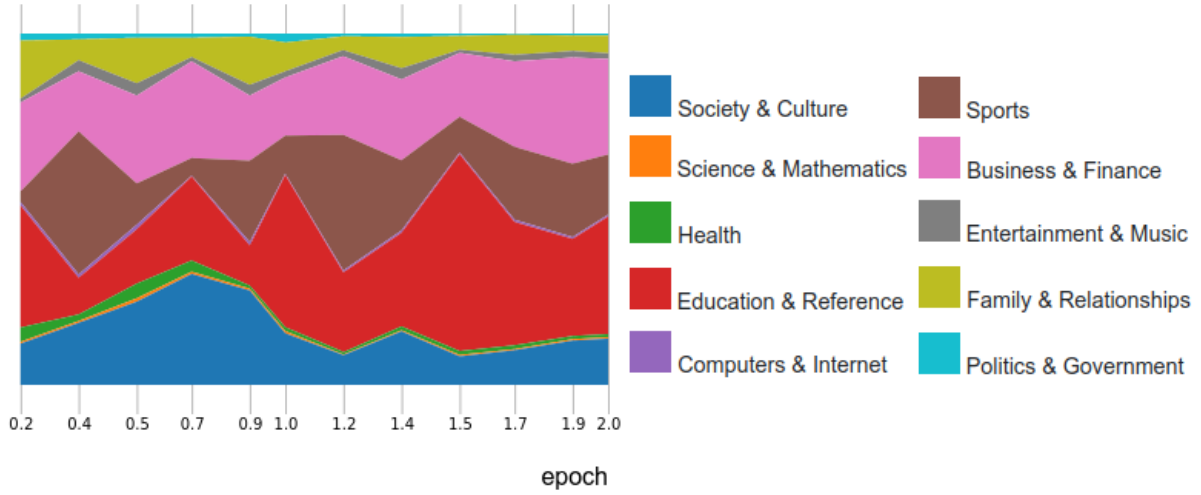


Figura 59 – Streamgraph, após a mitigação dos vieses, do caso onde o termo "community" era muito característico da classe "Education & Reference"

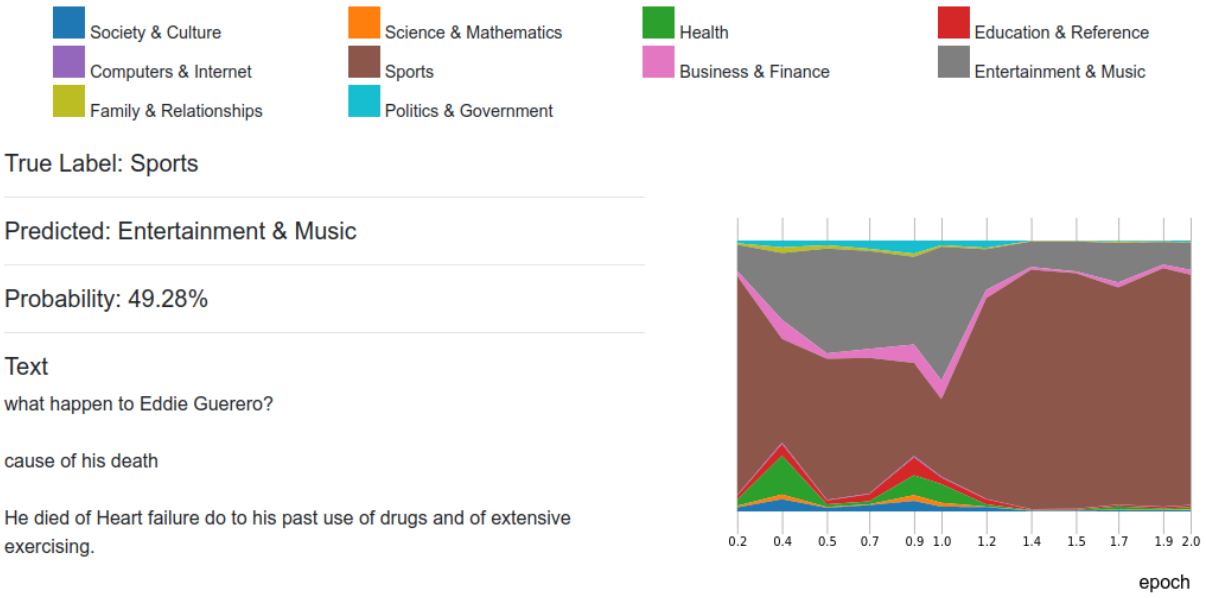
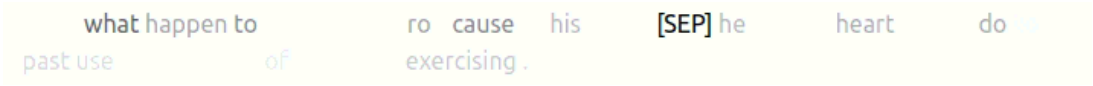


Figura 60 – Detalhes de uma instância rotulada como Sports, que ao longo do treinamento o modelo deixa de classificar como Entertainment & Music

Após a mitigação dos vieses, a importância da palavra "health" desaparece completamente, como mostra a Figura 71. Outros termos como "calories", "foods" e "fda" (a ANVISA americana) passaram a estar associados à classificação de "Health", que são termos afins com a classe.

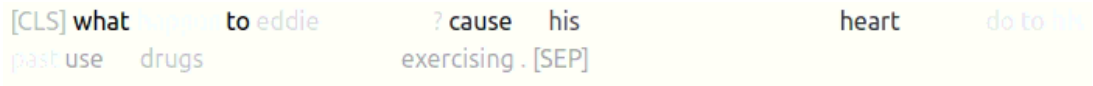
A Figura 72 mostra um caso com probabilidade próxima de 50% para a classe "Society & Culture", seguida por "Politics & Government", com probabilidade significativa ainda para a classe "Family & Relationships". As saliências mostram-se bem distintas para cada classe. Os termos "should gay couple" aparentam ser bem importantes para a classificação como "Society & Culture", "marriage" para a classe "Family & Relationships", e "australia" para "Politics



(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época

(b) Saliência para classificação de “Entertainment & Music” com o modelo treinado por 1 época

Figura 61 – Saliência do modelo treinado por uma época.

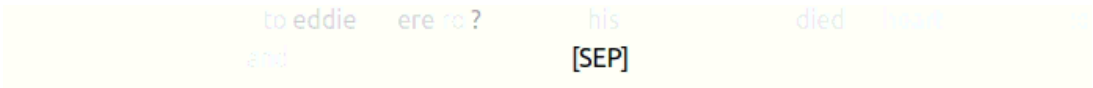


(a) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas

(b) Saliência para classificação de “Entertainment & Music” com o modelo treinado por 2 épocas

Figura 62 – Saliência do modelo treinado por duas épocas.

(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época, após a mitigação dos vieses



(b) Saliência para classificação de “Entertainment & Music” com o modelo treinado por 1 época, após mitigação dos vieses

(c) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas, após a mitigação dos vieses

(d) Saliência para classificação de “Entertainment & Music” com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 63 – Saliências do modelo treinado para mitigação dos vieses

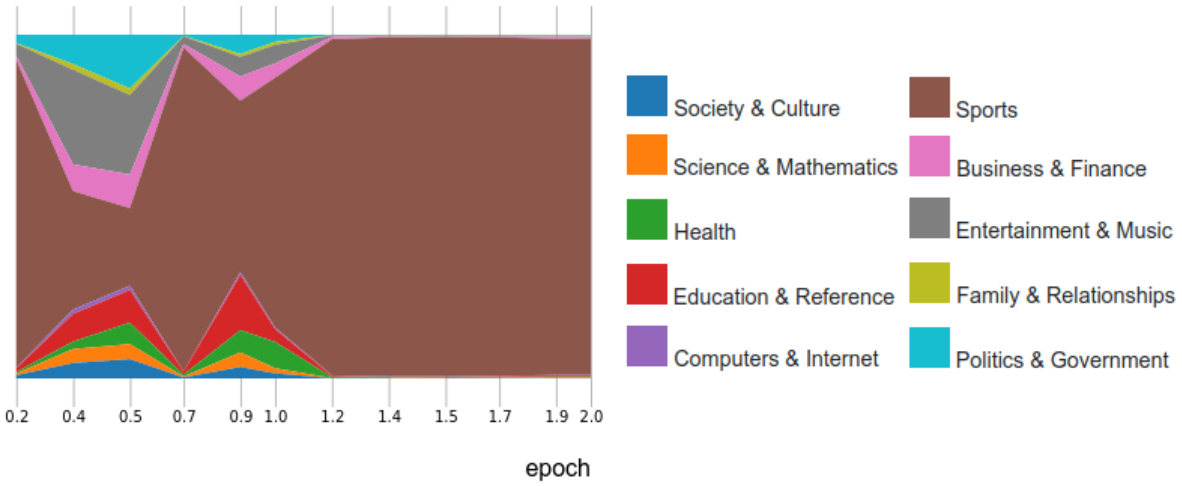


Figura 64 – Streamgraph, após a mitigação dos vieses, do caso onde o termo "Eddie" era muito característico da classe "Entertainment & Music"

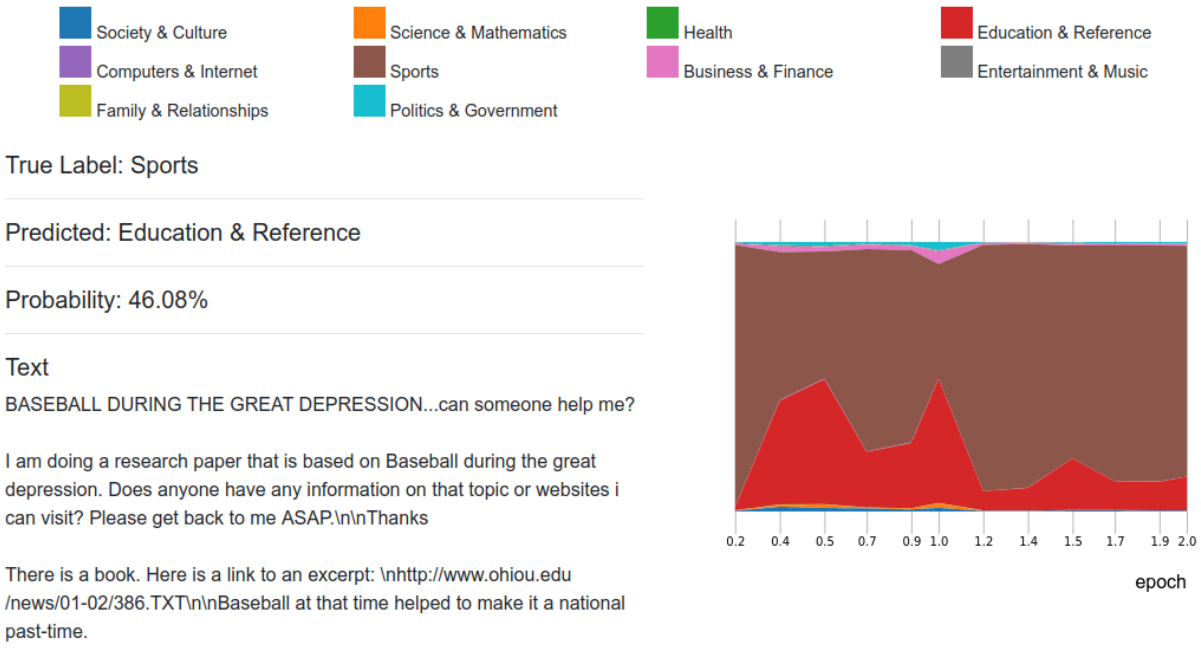


Figura 65 – Detalhes de uma instância rotulada como Sports, confundida com "Education & Reference" após uma época de treinamento

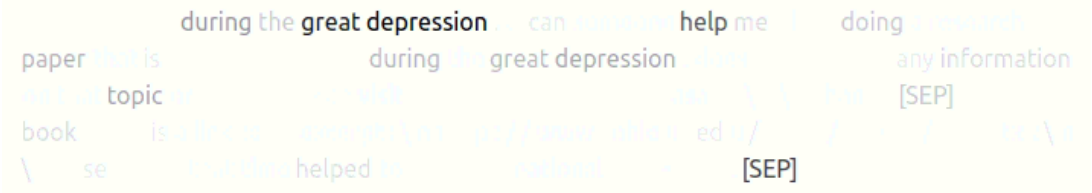
& Government". Destes três vieses, apenas "gay" trata-se de um viés não-intencional, pois também está relacionado com outras classes como "Family & Relationships" e "Health".

A Figura 73 mostra as saliências após a mitigação dos vieses. "couple" tornou-se uma palavra importante para a predição da classe "Society & Culture", ou seja, "gay" deixou de ser um viés neste caso. Portanto o viés não-intencional foi mitigado.

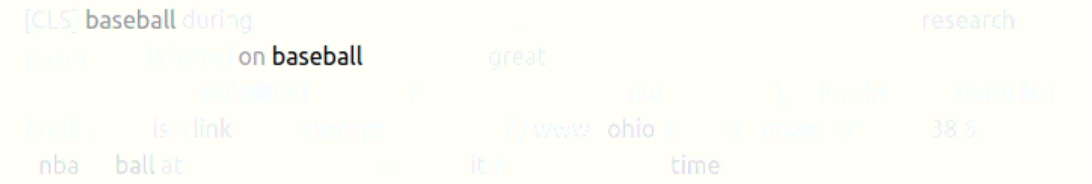
O streamgraph do modelo após a mitigação dos vieses, exibido na Figura 74, mostra que "Politics & Government" tornou-se a classe dominante.

A Figura 75 mostra outro caso com forte viés: o termo "homework" para classificação

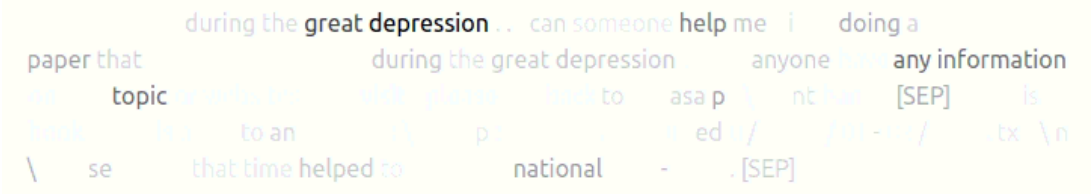
(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época



(b) Saliência para classificação de “Education & Reference” com o modelo treinado por 1 época

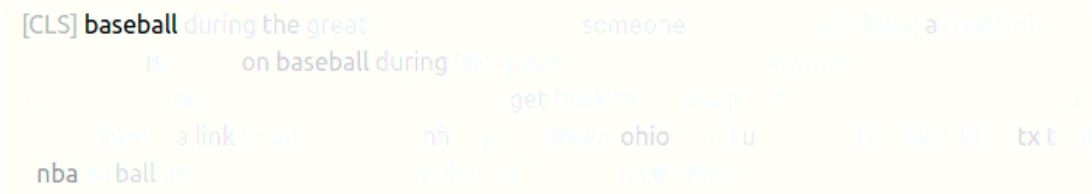


(c) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas

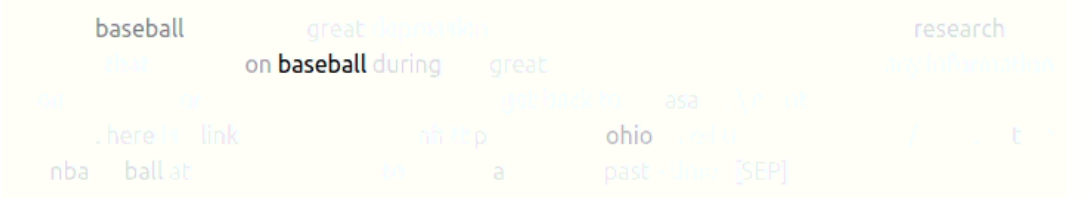


(d) Saliência para classificação de “Education & Reference” com o modelo treinado por 2 épocas

Figura 66 – Saliências do modelo treinado para mitigação dos vieses



(a) Saliência para classificação de “Sports” com o modelo treinado por 1 época



(b) Saliência para classificação de “Sports” com o modelo treinado por 2 épocas

Figura 67 – Saliências do modelo para a classe “Sports”.

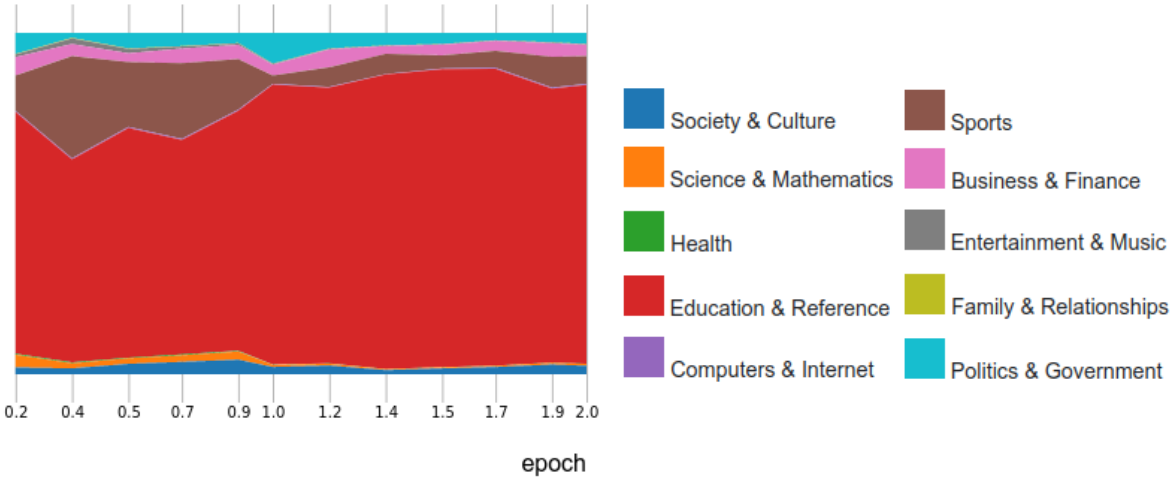


Figura 68 – Streamgraph, após a mitigação dos vieses, do caso onde o termo "baseball" era muito característico da classe "Sports"

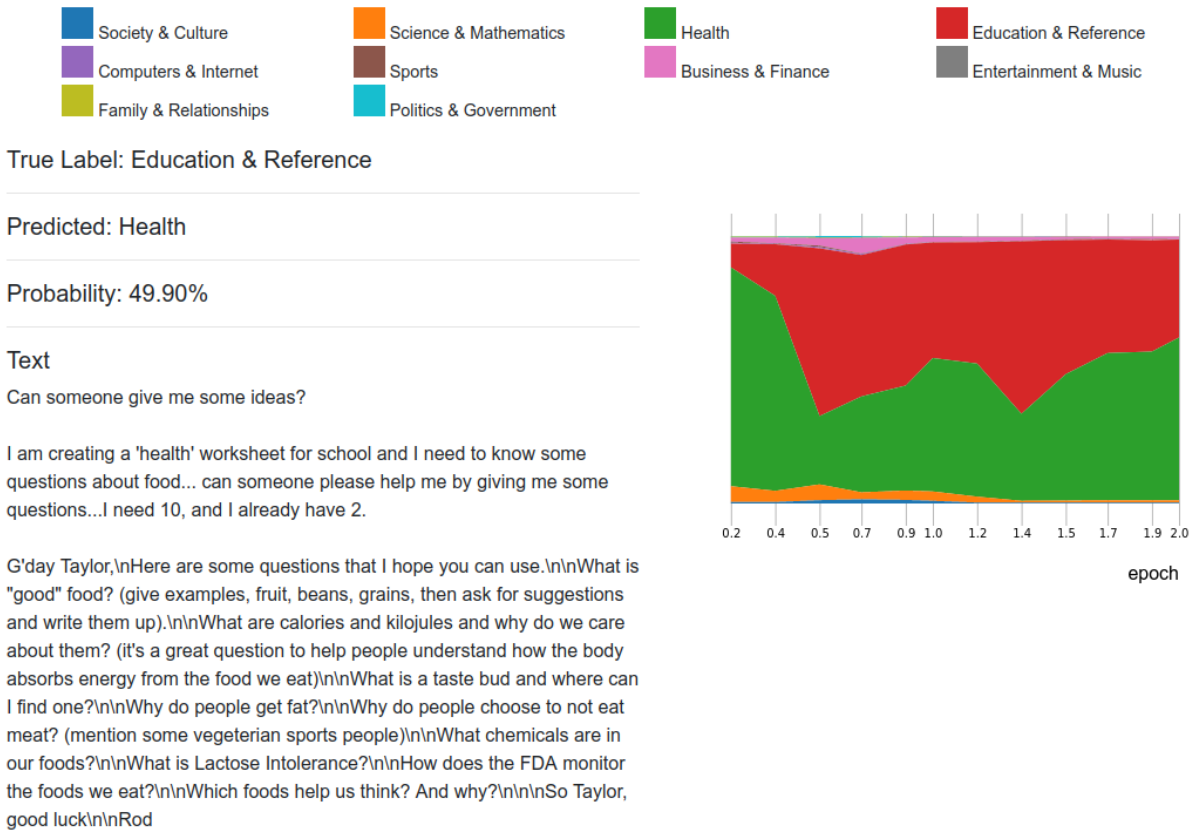
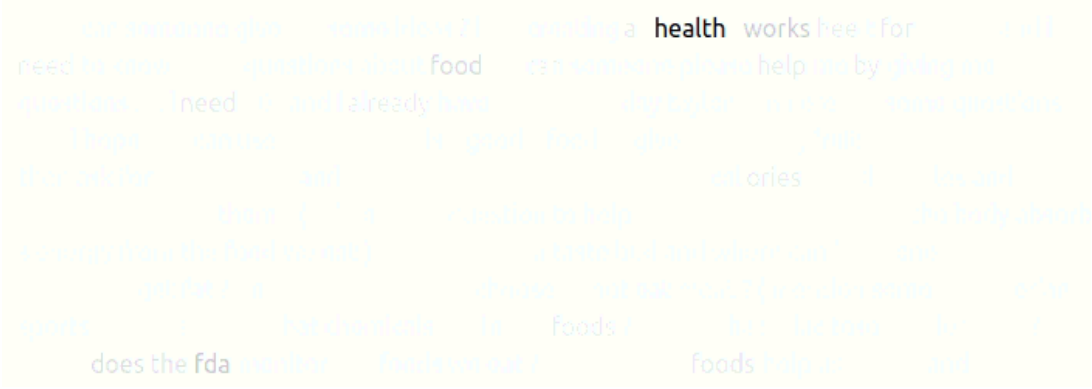


Figura 69 – Detalhes de uma instância para a qual o BERT está indeciso entre "Education & Reference" e "Health"

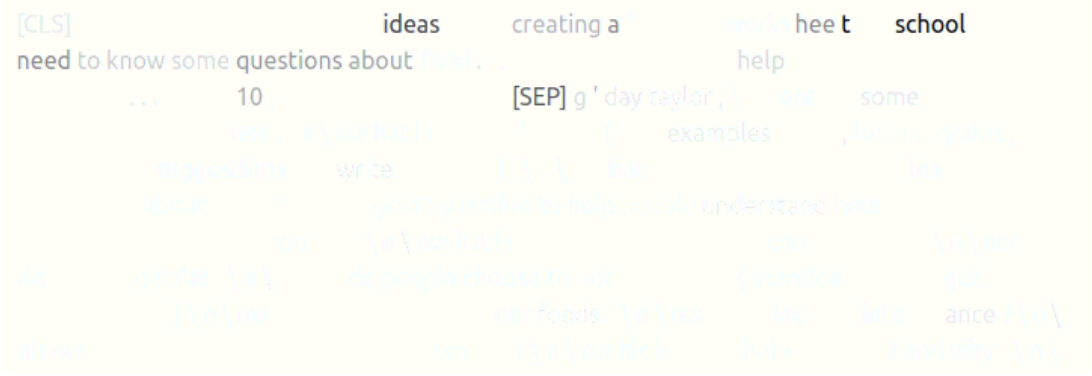
como "Education & Reference", a Figura 76 mostra as suas saliências. Antes de fazer a mitigação do viés, o modelo concentra sua decisão na palavra "homework" após treinar por mais uma época. Após a mitigação dos vieses, nota-se que o modelo se apoia em outras palavras relevantes, como "study", que desapareceu no modelo de duas épocas antes da mitigação. No entanto, a palavra "homework" perdeu sua importância, o que é negativo.



(a) Saliência para classificação de “Health” com o modelo treinado por 1 época

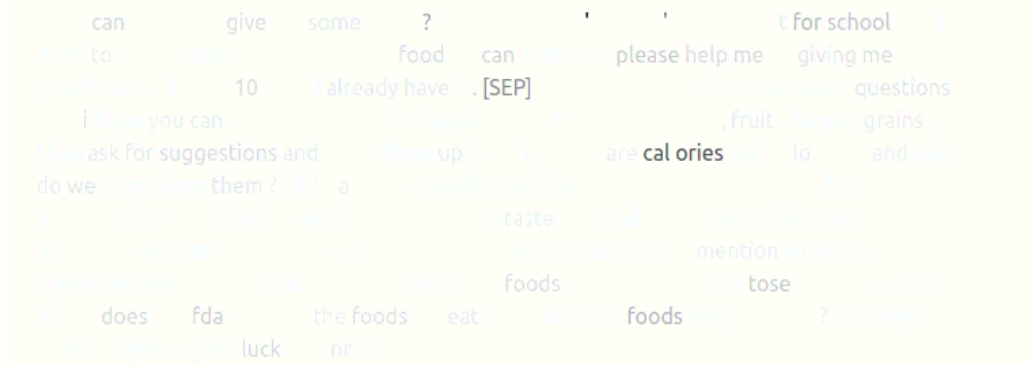
(b) Saliência para classificação de “Education & Reference” com o modelo treinado por 1 época

(c) Saliência para classificação de “Health” com o modelo treinado por 2 épocas



(d) Saliência para classificação de “Education & Reference” com o modelo treinado por 2 épocas

Figura 70 – Saliências do modelo treinado



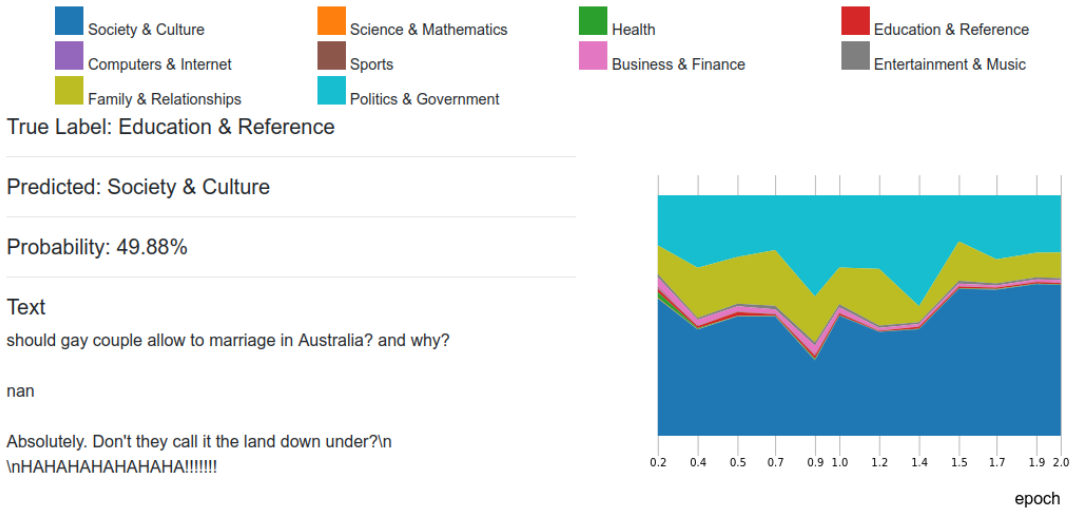
(a) Saliência para classificação de "Health" com o modelo treinado por 1 época, após a mitigação dos vieses

(b) Saliência para classificação de "Education & Reference" com o modelo treinado por 1 época, após a mitigação dos vieses

(c) Saliência para classificação de "Health" com o modelo treinado por 2 épocas, após a mitigação dos vieses

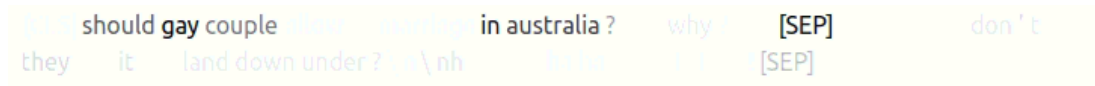
(d) Saliência para classificação de "Education & Reference" com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 71 – Saliências do modelo treinado, após a mitigação dos vieses

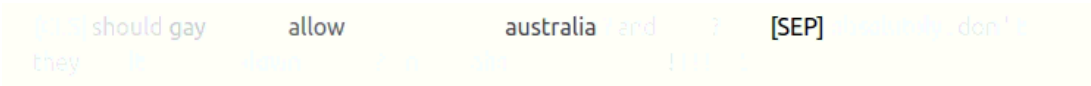


(a) Detalhes de uma instância para a qual o BERT está indeciso entre "Education & Reference" e "Health"

(b) Saliência para classificação de "Society & Culture" com o modelo treinado por 1 época



(c) Saliência para classificação de "Society & Culture" com o modelo treinado por 2 épocas



(d) Saliência para classificação de "Politics & Government" com o modelo treinado por 1 época

(e) Saliência para classificação de "Politics & Government" com o modelo treinado por 2 épocas

(f) Saliência para classificação de "Family & Relationships" com o modelo treinado por 1 época

(g) Saliência para classificação de "Family & Relationships" com o modelo treinado por 2 épocas

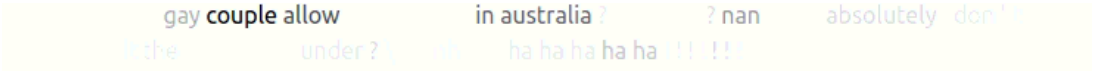
Figura 72 – Instância com saliências bem distintas entre as três classes principais

(a) Saliência para classificação de "Society & Culture" com o modelo treinado por 1 época, após a mitigação dos vieses

(b) Saliência para classificação de "Society & Culture" com o modelo treinado por 2 épocas, após a mitigação dos vieses

(c) Saliência para classificação de "Politics & Government" com o modelo treinado por 1 época, após a mitigação dos vieses

(d) Saliência para classificação de "Politics & Government" com o modelo treinado por 2 épocas, após a mitigação dos vieses



(e) Saliência para classificação de "Family & Relationships" com o modelo treinado por 1 época, após a mitigação dos vieses

(f) Saliência para classificação de "Family & Relationships" com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 73 – Instância com saliências bem distintas entre as três classes principais

A Figura 77 mostra o streamgraph após a mitigação dos vieses, onde nota-se que não ocorre o pico de probabilidade para a classe "Health". "Education & Reference" vai ganhando espaço de forma regular.

5.4 Análise quantitativa dos vieses observados

Na Tabela 4 estão organizados os termos mais salientes dentre os observados, juntamente com a classe predita pelo termo, e se trata-se de um viés intencional ou não-intencional. A palavra "baseball", por exemplo, é esperada como característica da classe "Sports", portanto trata-se de um viés intencional. Já a palavra "back", quando indicativa da parte das costas de um ser humano, não deveria ser uma característica da classe "Health", pois se enquadraria também para "Sports".

Para cada par temo-classe, a Tabela 4 mostra em quantos textos o termo aparece na classe (suporte), assim como a porcentagem correspondente na classe, comparando-a com a

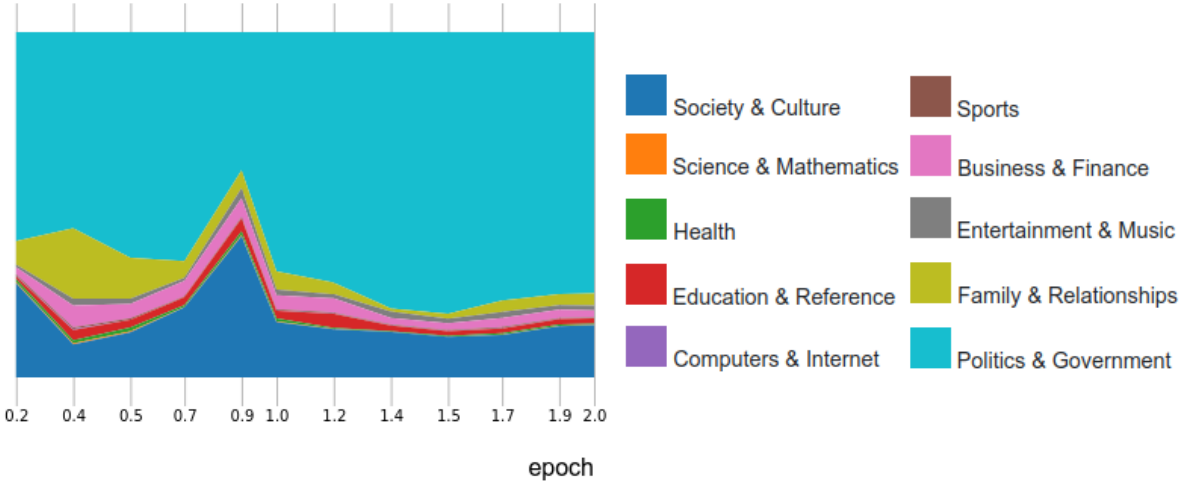


Figura 74 – Streamgraph, após a mitigação dos vieses, do caso onde o termo “gay” era muito característico da classe ”Society & Culture”

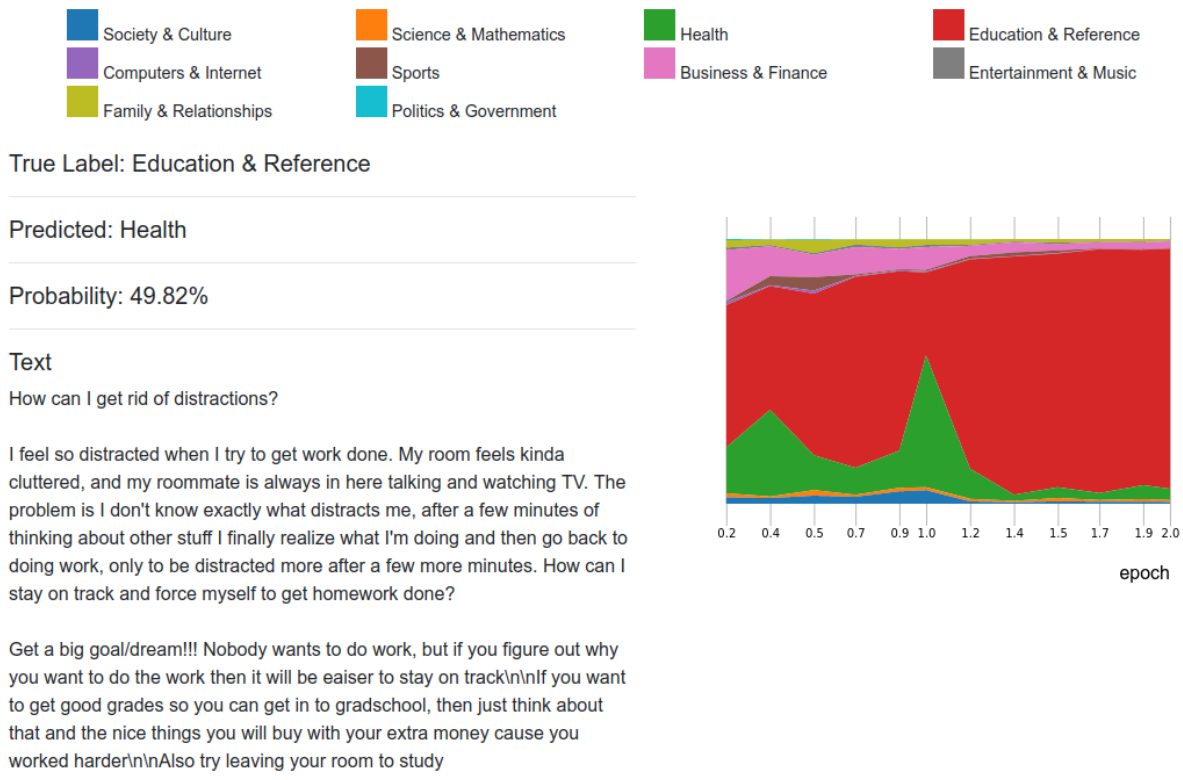
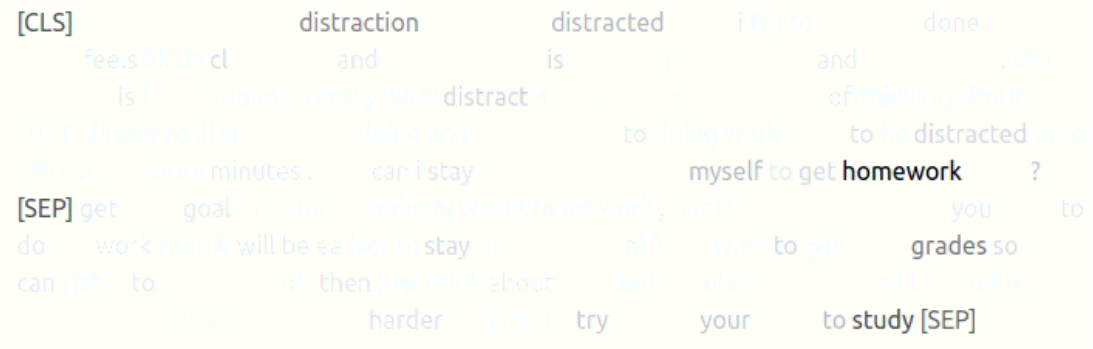


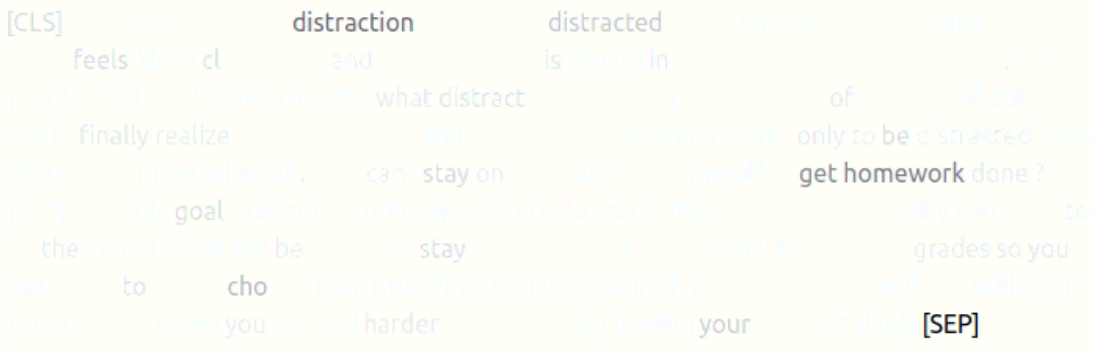
Figura 75 – Detalhes de uma instância

frequência no dataset. Dessa forma pode-se observar se o termo ocorre em maior proporção na classe. Todas as palavras salientes estão em maior proporção na classe investigada do que na média do dataset, como o esperado.

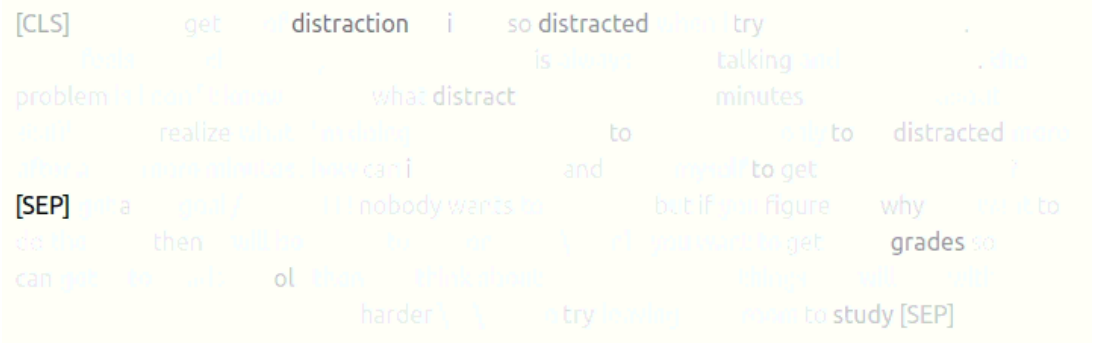
A Tabela 5 mostra as taxas de falsos positivos e falsos negativos associadas a cada termo. A palavra “thigh”, por exemplo, está associada a um terço dos falsos positivos da classe “Health”, e nenhum falso negativo.



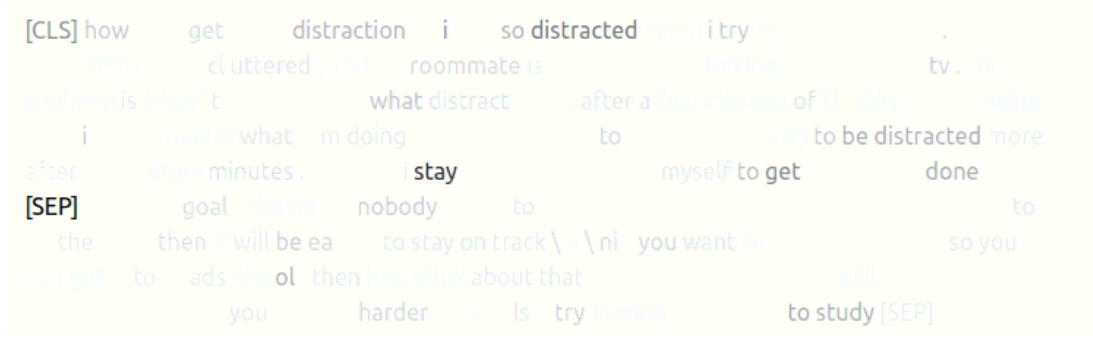
(a) Saliência para classificação de "Education & Reference" com o modelo treinado por 1 época



(b) Saliência para classificação de "Education & Reference" com o modelo treinado por 2 épocas



(c) Saliência para classificação de "Education & Reference" com o modelo treinado por 1 época, após a mitigação dos vieses



(d) Saliência para classificação de "Education & Reference" com o modelo treinado por 2 épocas, após a mitigação dos vieses

Figura 76 – Viés de "homework" para classificar como "Education & Reference"

Tabela 4 – Vieses observados e suas ocorrências

Termo	Classe	Suporte	Viés Intencional	Frequência no dataset	Frequência na classe
back	Health	517	Não	7,07%	10,02%
thigh	Health	19	Não	0,05%	0,30%
medicine	Health	128	Sim	0,48%	2,62%
data	Computer & Internet	262	Sim	0,82%	4,22%
baseball	Sports	248	Sim	0,51%	4,10%
802 (802.11)	Computers & Internet	16	Sim	0,02%	0,18%
teams	Sports	319	Sim	0,62%	5,61%
community	Education & Reference	89	Não	0,71%	1,57%
eddie	Entertainment & Music	9	Não	0,06%	0,22%
school	Education & Reference	836	Sim	3,80%	13,95%
health	Health	401	Não	1,30%	6,63%
marriage	Family & Relationships	216	Sim	0,79%	4,02%
gay	Society & Culture	190	Não	0,77%	2,85%
homework	Education & Reference	141	Sim	0,48%	2,19%

Tabela 5 – Taxa de falsos positivos e falsos negativos para cada par termo-classe

Termo	Classe	FPR (1 época)	FPR (2 épocas)	FNR (1 época)	FNR (2 épocas)
back	Health	2,49%	1,78%	8,32%	9,28%
thigh	Health	33,33%	33,33%	0%	0%
medicine	Health	28,13%	23,44%	7,03%	7,81%
data	Computer & Internet	11,52%	9,95%	1,91%	3,44%
baseball	Sports	18,75%	16,67%	1,61%	2,02%
802 (802.11)	Computers & Internet	28,57%	28,57%	0%	0%
teams	Sports	19,51%	14,63%	0,63%	0%
community	Education & Reference	6,61%	8,68%	31,46%	21,35%
eddie	Entertainment & Music	9,67%	6,45%	0%	0%
school	Education & Reference	12,44%	13,52%	15,55%	13,16%
health	Health	24,06%	20,00%	5,49%	7,98%
marriage	Family & Relationships	16,47%	19,41%	8,80%	9,26%
gay	Society & Culture	27,84%	25,43%	15,79%	15,26%
homework	Education & Reference	12,59%	18,18%	29,79%	15,60%

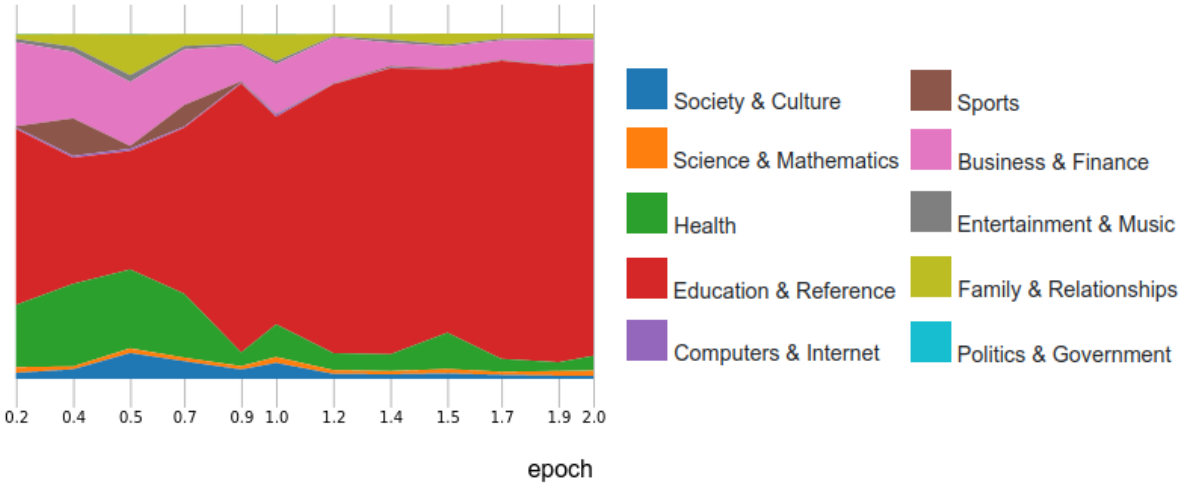


Figura 77 – Streamgraph, após a mitigação dos vieses, do caso onde o termo "homework" era muito característico da classe "Education & Reference"

Para analisar se os vieses atrapalham o resultado da classificação, é preciso comparar com a média do dataset. Para fazer essa comparação, foram calculados os FPRs e FNRs de cada classe, e então subtraídos dos FPRs e FNRs de cada termo em relação à classe de interesse. Os resultados estão na Tabela 6, onde números positivos significam que a taxa de erros é maior no dataset como um todo, e número negativos significam que ela é pior nos casos com o termo. De qualquer forma, quanto mais próximo de zero, menor o viés associado ao termo.

O termo "thigh" apresenta um diferença nos falsos positivos de -30,53%, mostrando que este termo é fortemente responsável pelos falsos positivos em que aparece. Outro caso com alta diferença de FPRs negativa o termo "medicine", que causa muitos erros deste tipo na classe "Health". Dentre os falsos negativos destaca-se o termo "school", que, quando presente, leva à classificação como "Education & Reference" 35% a mais do que para as outras classes, quando o rótulo é outro.

5.5 Mitigação dos vieses

Primeiramente contabilizou-se a ocorrência dos tokens de interesse em todas as classes, com exceção da classe investigada. Um dos pares de interesse, por exemplo, é a palavra "back" na classe "Health". Originalmente, a palavra "back" aparece em 79.239 textos, sendo 11.219 na classe "Health" e 68.020 nas demais. Para eliminar o viés, apenas 7.557 textos da "Health" que contém a palavra "back" foram mantidos no dataset, e os demais foram descartados. Dessa forma, a classe "Health" contém 10% dos textos com a palavra "back".

A Tabela 7 mostra a quantidade das palavras enviesadas no dataset original, e também depois da equalização para redução dos vieses. Nota-se que a quantidade final nas classes fica aproximadamente 10% do que consta no dataset.

Tabela 6 – Diferença de FPRs e FNRs para cada termo nas classes de interesse

Termo	Classe	Diferença de FPRs (1 época)	Diferença de FNRs (1 época)
back	Health	0,31%	4,97%
thigh	Health	-30,53%	13,28%
medicine	Health	-25,32%	6,25%
data	Computer & Internet	-9,50%	5,92%
baseball	Sports	-18,09%	6,42%
802 (802.11)	Computers & Internet	-26,55%	7,83%
teams	Sports	-18,85%	7,41%
community	Education & Reference	-4,61%	19,06%
eddie	Entertainment & Music	-7,13%	20,78%
school	Education & Reference	-10,44%	34,97%
health	Health	-21,26%	7,80%
marriage	Family & Relationships	-14,15%	11,75%
gay	Society & Culture	-22,92%	13,73%
homework	Education & Reference	-10,59%	20,73%

Tabela 7 – Ocorrência dos termos no dataset e na classe de interesse, antes e depois do preparo para a mitigação dos vieses

Termo	Classe	Ocorrência do termo no dataset original	Ocorrência do termo na classe (dataset original)	Ocorrência do termo nas demais classes (dataset original)	Ocorrência do termo na classe após filtragem
back	Health	79239	11219	68020	7557
thigh	Health	530	333	197	21
medicine	Health	5327	2936	2391	265
data	Computer & Internet	9236	4730	4506	500
baseball	Sports	5706	4591	1115	123
802 (802.11)	Computers & Internet	265	207	58	6
teams	Sports	6938	6288	650	72
community	Education & Reference	7896	1763	6133	681
eddie	Entertainment & Music	649	251	398	44
school	Education & Reference	42530	15619	26911	2990
health	Health	14518	7429	7089	787
marriage	Family & Relationships	8893	4499	4394	488
gay	Society & Culture	8567	3191	5376	597
homework	Education & Reference	5350	2448	2902	322

Tabela 8 – Comparação entre FPRs e FNRs dos modelos treinados com o dataset Yahoo Answers padrão e o treinado com o dataset preparado para mitigação dos vieses. Apenas os modelos treinados por uma época são apresentados

Termo	Classe	FPR	FPR Mitigado	FNR	FNR Mitigado
back	Health	2,49%	2,11%	8,32%	10,64%
thigh	Health	33,33%	33,33%	0%	0%
medicine	Health	28,13%	14,06%	7,03%	17,97%
data	Computer & Internet	11,52%	5,76%	1,91%	4,58%
baseball	Sports	18,75%	8,33%	1,61%	9,27%
802 (802.11)	Computers & Internet	28,57%	28,75%	0%	0%
teams	Sports	19,51%	17,07%	0,63%	0,94%
community	Education & Reference	6,61%	7,44%	31,46%	28,09%
eddie	Entertainment & Music	9,67%	6,45%	0%	0%
school	Education & Reference	12,44%	7,37%	15,55%	27,39%
health	Health	24,06%	8,12%	5,49%	21,70%
marriage	Family & Relationships	16,47%	8,82%	8,80%	16,67%
gay	Society & Culture	27,84%	7,56%	15,79%	47,37%
homework	Education & Reference	12,59%	9,79%	29,79%	39,01%

A Tabela 8 compara as FPRs e FNRs do modelo treinado com o dataset normal contra o dataset treinado para mitigação dos vieses. Nota-se que a taxa de falsos positivos caiu muito na maioria dos casos, porém a taxa de falsos negativos subiu. Ou seja, o modelo aponta menos casos errados para a maioria dos vieses, mas deixa de reconhecer muitos outros. Essa queda de desempenho nos FNRs pode ser um reflexo da estratégia de mitigação, afinal ao retirar textos do treinamento o modelo pode ter deixado de aprender outras características importantes sobre as classes.

Para fins de registro, a Tabela 9 apresenta os FPRs e FNRs dos termos para modelo após a mitigação dos vieses.

Tabela 9 – Taxas de falsos positivos e falsos negativos dos termos analisados, para as classes investigadas, no modelo após mitigação dos vieses

Termo	Classe	Diferença de falsos positivos (1 época)	Diferença de falsos positivos (2 épocas)	Diferença de falsos negativos (1 época)	Diferença de falsos negativos (2 épocas)
back	Health	0,50%	0,43%	4,68%	4,77%
thigh	Health	-30,73%	-31,12%	15,32%	17,53%
medicine	Health	-11,46%	-4,03%	-2,65%	-22,31%
data	Computer & Internet	-4,36%	-4,10%	6,38%	-0,61%
baseball	Sports	-7,62%	-5,49%	-1,07%	-5,54%
802 (802.11)	Computers & Internet	-27,17%	-26,91%	10,97%	9,32%
teams	Sports	-16,36%	-11,44%	7,26%	5,57%
community	Education & Reference	-4,38%	-2,82%	16,74%	11,92%
eddie	Entertainment & Music	-3,96%	-3,78%	20,90%	20,62%
school	Education & Reference	-4,31%	-2,11%	17,44%	8,83%
health	Health	-5,52%	-4,04%	-6,38%	-10,65%
marriage	Family & Relationships	-5,09%	-1,15%	-2,08%	-9,21%
gay	Society & Culture	-4,07%	-5,41%	-10,79%	-12,43%
homework	Education & Reference	-6,73%	-4,03%	5,83%	-0,06%

6 Conclusões e Trabalhos Futuros

Garantir a qualidade das anotações em datasets preparados para o treinamento de modelos de aprendizado supervisionado, como o BERT, é primordial para a obtenção de um bom desempenho e também para sua correta avaliação. O presente trabalho traz duas contribuições neste sentido: i) a ferramenta Mapa de Instâncias, que facilita a inspeção de exemplos do conjunto de treinamento ou teste, possibilitando uma revisão eficiente dos rótulos do dataset; e ii) a aplicação da visualização por saliência, conforme a técnica Integrated Gradients, para o estudo de características aprendidas pelo classificador, que muitas vezes são reflexo de vieses presentes no dataset de treinamento.

6.1 Conclusões

O Mapa de Instâncias permite uma visualização eficiente de um conjunto de dados, representado-o de forma compacta na tela, além de uma boa interatividade para efetuar-se uma exploração caso a caso. Torna-se uma alternativa interessante à análise sequencial do dataset, pois fornece uma visão geral do conjunto inteiro, facilitando a navegação para os casos de maior interesse.

Aplicando o ranqueamento das instâncias de acordo com a probabilidade da classe predita, e colorindo-as de acordo com o rótulo, o Mapa de Instâncias mostrou-se útil para a tarefa de revisão de anotações. A quantidade de casos mal rotulados foi o dobro da encontrada por seleção aleatória no topo do mapa de cada classe. E uma segunda capacidade observada é a detecção de textos mal escritos ou que não pertencem à nenhuma das classes definidas no dataset, estes dispostos na base dos mapas.

O uso do Streamgraph para visualização da distribuição de probabilidades ao longo do treinamento permite a identificação de diferentes tipos de instâncias: casos ambíguos, textos mal escritos, e casos característicos de uma classe. Para os casos ambíguos a probabilidade de classificação fica oscilando entre duas classes em diferentes checkpoints do modelo, sem a dominância de nenhuma classe. Para os casos mal escritos, nenhuma classe se destaca ao longo do treinamento. Já para os casos característicos de uma classe, o streamgraph torna-se praticamente monocromático.

O Integrated Gradients mostrou-se capaz de identificar o overfitting na comparação entre o modelo treinado por épocas diferentes. Após a mitigação dos vieses, o modelo se apoia em mais características para efetuar suas predições, tornando-se, portanto, mais robusto. Essas observações qualitativas concordam com as medidas quantitativas de taxas de falsos positivos, e também com a loss no conjunto de validação medida durante o treinamento do modelo.

Tomar a decisão do melhor checkpoint baseando-se apenas nas métricas comuns — precisão, recall e f1-score — levaria à escolha do modelo com severo overfitting.

Outra observação interessante feita por meio do Integrated Gradients está relacionada com as classes ambíguas, onde as partes salientes estão de acordo com o que se espera de cada classe. Isso mostra que a aplicação desta técnica é válida para avaliar-se a qualidade do BERT treinado.

6.2 Trabalhos futuros

Comparar a capacidade do emprego do Mapa de Instâncias para a detecção de casos mal rotulados utilizando outros modelos além do BERT, a fim de avaliar sua eficiência para esta tarefa. O uso de modelos lineares, por exemplo, pode ser complementar e revelar outros tipos de erros de anotação no topo do mapa de cada classe.

Outra aplicação interessante para o Mapa de Instâncias é a tarefa de Active Learning, na qual o usuário anota um dataset não rotulado de acordo com as predições de um modelo já treinado, geralmente priorizando os casos com maior indecisão. A comparação do tempo despendido em softwares já existentes para essa tarefa, contra o tempo despendido usando o mapa de instâncias, pode revelá-lo como uma boa alternativa.

O experimento do uso do mapa de instâncias como ferramenta para a detecção de casos mal rotulados deve ser ampliado, com a observação de mais casos para obter-se maior significância estatística nas conclusões. A aplicação em um segundo dataset também é interessante para corroborar os resultados.

Quanto ao Integrated Gradients, em muitas visualizações por saliência o token separador ([SEP]) fica muito saliente, apagando todas as outras palavras, o que torna difícil sua interpretação. Uma continuidade interessante do trabalho seria avaliar o resultado ao ignorar a saliência para este token, o que deve ser avaliado em conjunto com o token "[CLS]".

Para complementar o estudo da mitigação dos vieses, uma segunda estratégia de mitigação pode ser aplicada, na qual textos são adicionados para equilibrar a proporção dos tokens entre a classe alvo e o dataset, e não removidos como foi feito. Espera-se que a taxa de falsos positivos mantenha-se menor do que com o dataset original, mas sem aumento na taxa de falsos negativos. Contudo é difícil encontrar textos que possam ser rotulados com as mesmas classes do dataset Yahoo Answers.

Por fim, torna-se interessante utilizar a saliência na camada de entrada do BERT como um ponto inicial para entender os seus mecanismos internos, tentando entender como os diferentes tokens salientes interagem entre si até a camada final da rede neural. Ou seja, complementar o presente trabalho com o estudo das representações de camadas intermediárias do BERT, ou mesmo com a análise dos mecanismos de atenção.

Referências

ABNAR, S.; ZUIDEMA, W. Quantifying attention flow in transformers. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020. p. 4190–4197. Disponível em: <https://www.aclweb.org/anthology/2020.acl-main.385>.

Adadi, A.; Berrada, M. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, v. 6, p. 52138–52160, 2018.

AKEN, B. v.; WINTER, B.; LöSER, A.; GERS, F. A. Visbert: Hidden-state visualizations for transformers. In: *Companion Proceedings of the Web Conference 2020*. New York, NY, USA: Association for Computing Machinery, 2020. p. 207–211. ISBN 9781450370240. Disponível em: <https://doi.org/10.1145/3366424.3383542>.

AROCA-OUELLETTE, S.; RUDZICZ, F. On Losses for Modern Language Models. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. p. 4970–4981. Disponível em: <https://www.aclweb.org/anthology/2020.emnlp-main.403>.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

BRASOVEANU, A. M.; ANDONIE, R. Visualizing transformers for nlp: A brief survey. *24th International Conference Information Visualisation (IV)*, 2020.

BYRON, L.; WATTENBERG, M. Stacked graphs – geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics*, v. 14, n. 6, p. 1245–1252, 2008.

CLARK, K.; KHANDELWAL, U.; LEVY, O.; MANNING, C. D. What does bert look at? an analysis of bert's attention. In: *BlackBoxNLP@ACL*. [S.l.: s.n.], 2019.

DANILEVSKY, M.; QIAN, K.; AHARONOV, R.; KATSIS, Y.; KAWAS, B.; SEN, P. A survey of the state of explainable AI for natural language processing. In: *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. Suzhou, China: Association for Computational Linguistics, 2020. p. 447–459. Disponível em: <https://www.aclweb.org/anthology/2020.aacl-main.46>.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: *CVPR09*. [S.l.: s.n.], 2009.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

DIXON, L.; LI, J.; SORENSEN, J.; THAIN, N.; VASSERMAN, L. Measuring and mitigating unintended bias in text classification. In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. New York, NY, USA: Association for Computing Machinery, 2018. (AIES '18), p. 67–73. ISBN 9781450360128. Disponível em: <https://doi.org/10.1145/3278721.3278729>.

EBRAHIMI, J.; RAO, A.; LOWD, D.; DOU, D. HotFlip: White-box adversarial examples for text classification. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, 2018. p. 31–36. Disponível em: <https://www.aclweb.org/anthology/P18-2006>.

FERRI, C.; HERNáNDEZ-ORALLO, J.; MODROIU, R. An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, v. 30, n. 1, p. 27–38, 2009. ISSN 0167-8655. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167865508002687>.

GOLDBERG, Y. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, v. 57, p. 345–420, 2016.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

HOOVER, B.; STROBELT, H.; GEHRMANN, S. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, 2020. p. 187–196. Disponível em: <https://www.aclweb.org/anthology/2020.acl-demos.22>.

KEIM, D. A. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on visualization and computer graphics*, IEEE, v. 6, n. 1, p. 59–78, 2000.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, v. 25, p. 1097–1105, 2012.

LEE, J.; YOON, W.; KIM, S.; KIM, D.; KIM, S.; SO, C. H.; KANG, J. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, Oxford University Press, v. 36, n. 4, p. 1234–1240, 2020.

LIN, J. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, IEEE, v. 37, n. 1, p. 145–151, 1991.

LIU, Y.; OTT, M.; GOYAL, N.; DU, J.; JOSHI, M.; CHEN, D.; LEVY, O.; LEWIS, M.; ZETTLEMOYER, L.; STOYANOV, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G.; DEAN, J. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.

MILLER, T.; HOWE, P.; SONENBERG, L. Explainable AI: Beware of inmates running the asylum. In: *IJCAI 2017 Workshop on Explainable Artificial Intelligence (XAI)*. [s.n.], 2017. Disponível em: <http://people.eng.unimelb.edu.au/tmiller/pubs/explanation-inmates.pdf>.

MOTAMEDI, M.; SAKHARNYKH, N.; KALDEWEY, T. A data-centric approach for training deep neural networks with less data. *arXiv preprint arXiv:2110.03613*, 2021.

NGUYEN, K.; FOOKES, C.; ROSS, A.; SRIDHARAN, S. Iris recognition with off-the-shelf cnn features: A deep learning perspective. *IEEE Access*, PP, p. 1–1, 12 2017.

PAIVA, P. Y. A.; SMITH-MILES, K.; VALERIANO, M. G.; LORENA, A. C. Pyhard: a novel tool for generating hardness embeddings to support data-centric analysis. *arXiv preprint arXiv:2109.14430*, 2021.

POLYZOTIS, N.; ZAHARIA, M. What can data-centric ai learn from data and ml engineering? *arXiv preprint arXiv:2112.06439*, 2021.

RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. Language models are unsupervised multitask learners. 2019.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.

RUSSELL STUART J, S. J. *Inteligência artificial*. Rio de Janeiro: Elsevier. ISBN 9788535237016.

SIMONYAN, K.; VEDALDI, A.; ZISSERMAN, A. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014.

SMILKOV, D.; THORAT, N.; KIM, B.; VIéGAS, F.; WATTENBERG, M. *SmoothGrad: removing noise by adding noise*. 2017.

STRICKLAND, E. Andrew ng, ai minimalist: The machine-learning pioneer says small is the new big. *IEEE Spectrum*, v. 59, n. 4, p. 22–50, 2022.

SUNDARARAJAN, M.; TALY, A.; YAN, Q. Axiomatic attribution for deep networks. In: *Proceedings of the 34th International Conference on Machine Learning - Volume 70*. [S.l.]: JMLR.org, 2017. (ICML'17), p. 3319–3328.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 5998–6008.

VIG, J. A multiscale visualization of attention in the transformer model. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Florence, Italy: Association for Computational Linguistics, 2019. p. 37–42. Disponível em: <https://www.aclweb.org/anthology/P19-3007>.

WALLACE, E.; TUYLS, J.; WANG, J.; SUBRAMANIAN, S.; GARDNER, M.; SINGH, S. AllenNLP interpret: A framework for explaining predictions of NLP models. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*. Hong Kong, China: Association for Computational Linguistics, 2019. p. 7–12. Disponível em: <https://www.aclweb.org/anthology/D19-3002>.

WU, Y.; SCHUSTER, M.; CHEN, Z.; LE, Q. V.; NOROUZI, M.; MACHEREY, W.; KRIKUN, M.; CAO, Y.; GAO, Q.; MACHEREY, K.; KLINGNER, J.; SHAH, A.; JOHNSON, M.; LIU, X.; KAISER Łukasz; GOUWS, S.; KATO, Y.; KUDO, T.; KAZAWA, H.; STEVENS, K.; KURIAN, G.; PATIL, N.; WANG, W.; YOUNG, C.; SMITH, J.; RIESA, J.; RUDNICK, A.; VINYALS,

O.; CORRADO, G.; HUGHES, M.; DEAN, J. *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016.

YANG, Z.; DAI, Z.; YANG, Y.; CARBONELL, J.; SALAKHUTDINOV, R. R.; LE, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2019. p. 5753–5763.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. In: FLEET, D.; PAJDLA, T.; SCHIELE, B.; TUYTELAARS, T. (Ed.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014. p. 818–833. ISBN 978-3-319-10590-1.

ZHANG, X.; ZHAO, J.; LECUN, Y. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, v. 28, 2015.

Apêndices

APÊNDICE A – Dataset utilizado e treinamento do modelo

A.1 O dataset Yahoo Answers

O dataset Yahoo Answers (ZHANG; ZHAO; LECUN, 2015) é composto por pares de perguntas e respostas, extraídos automaticamente da plataforma homônima, na qual os usuários podiam fazer perguntas em diferentes categorias, enquanto outras pessoas as respondiam. É um dataset grande, com um milhão e 400 mil textos de treinamento, e um conjunto de 60 mil textos já separados para teste, perfeitamente balanceados entre dez classes. Vinte porcento dos textos de treinamento foram aleatoriamente selecionados para a validação do modelo durante o treinamento, respeitando a proporção em relação às classes. A Tabela 10 mostra a lista de classes do dataset e as quantidades de cada uma destinadas para treinamento, validação e teste.

O dataset contém três campos textuais, além do rótulo: i) o título da pergunta; ii) o conteúdo da pergunta, que em muitos casos está vazio, ou seja, a pergunta consiste somente no título; e iii) a resposta. Um exemplo de instância do dataset, pertencente à categoria "Sports", é a pergunta de título "Will Pakistan beat India in cricket matches?", com resposta "May Be or May be Not. But I hope the Cricket Matches will be interesting. Whether Parkistan win or India. The Love and Friendship between Two Must Win". Este exemplo possui o campo de conteúdo da pergunta vazio.

Tabela 10 – Classes do dataset Yahoo Answers e suas quantidades nos conjuntos de treinamento, validação e teste

Classe	Exemplos de treinamento	Exemplos de validação	Exemplos de teste
Society & Culture	112000	28000	10000
Science & Mathematics	112000	28000	10000
Health	112000	28000	10000
Education & Reference	112000	28000	10000
Computer & Internet	112000	28000	10000
Sports	112000	28000	10000
Business & Finance	112000	28000	10000
Entertainment & Music	112000	28000	10000
Family & Relationships	112000	28000	10000
Politics & Government	112000	28000	10000

A.2 Detalhes do treinamento

O BERT pré-treinado utilizado foi o "bert-base-uncased" (DEVLIN et al., 2018), que então passou por fine-tuning para a tarefa de classificação dos textos por duas épocas completas no dataset. Para acomodar o dataset ao formato de entrada do BERT, o título e o conteúdo da pergunta foram unidos e colocados no "segmento A" da entrada, conforme descrito na Seção 17, e a resposta no "segmento B".

O modelo foi treinado em uma GeForce GTX 1070 Ti, com os seguintes hiper-parâmetros: i) entrada máxima de 256 subtokens; ii) batch size de 16 exemplos; iii) taxa de aprendizado inicial de 2e-5; e iv) decaimento linear da taxa de aprendizado, tornando-se zero ao final das duas épocas.

Por tratar-se de um dataset grande, o modelo foi avaliado a cada 200 mil exemplos (correspondentes a 12500 atualizações de pesos), resultando em seis checkpoints por época. A Figura 78 mostra a curva de aprendizado do modelo. O menor erro no conjunto de validação ocorre na época "0,89", ficando bem próximo do obtido após uma época completa do treinamento. Nota-se o overfitting profundo alcançado pelo modelo da primeira época em diante, sem benefício no desempenho em relação ao conjunto de validação.

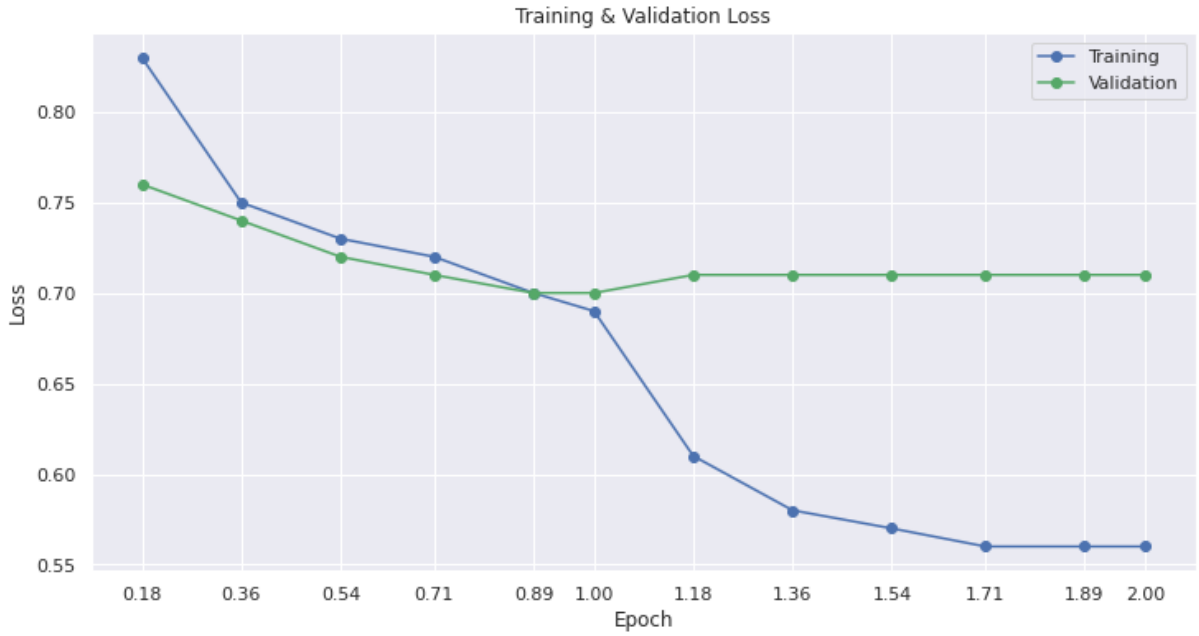


Figura 78 – Curva de aprendizado do BERT treinado, conforme a loss de entropia cruzada. Os valores de treinamento referem-se à loss média nos últimos 200 mil exemplos vistos, enquanto a validação é sempre feita no conjunto completo de validação

A.3 Resultados obtidos

A Tabela 11 mostra o desempenho obtido pelo modelo conforme o conjunto de teste e o de validação, separado entre o modelo treinado por uma época e o modelo que passou

por duas épocas de treinamento. Nota-se que o modelo que sofre com o overfitting atinge um desempenho melhor do que o modelo treinado por apenas uma época, mesmo que sua loss seja pior. Isso leva a crer que a melhoria no resultado de classificação é apoiada pelo uso de viéses do dataset, não refletindo um melhor aprendizado de fato, o que é corroborado pelas observações feitas na Seção 5.

Tabela 11 – Métricas gerais: F1-Score obtido pelo BERT após 1 e também após 2 épocas de treinamento, avaliados nos conjuntos de teste e de validação

Conjunto	Épocas de Treinamento	F1-Score
Teste	1	76,85
Teste	2	77,33
Validação	1	76,66
Validação	2	77,14

As subseções a seguir detalham o resultado de cada modelo por classe.

A.3.1 Modelo treinado por 1 época

A.3.1.1 Avaliação no conjunto de teste

Os resultados obtidos pelo modelo, após 1 época completa de treinamento, avaliados no conjunto de teste, estão na Tabela 12. A classe mais fácil para o modelo é "Sports", seguida por "Computer & Internet", enquanto as mais problemáticas são "Education & Reference" e "Business & Finance".

Tabela 12 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de teste

Classe	Precisão	Recall	F1-Score
Society & Culture	61.95	70.77	66.07
Science & Mathematics	73.40	84.40	78.52
Health	77.24	86.90	81.79
Education & Reference	73.16	50.62	59.84
Computers & Internet	84.34	92.07	88.03
Sports	93.78	92.25	93.01
Business & Finance	70.41	52.87	60.39
Entertainment & Music	78.27	79.08	78.68
Family & Relationships	79.86	79.35	79.60
Politics & Government	80.22	85.10	82.59

A Figura 79 apresenta a matriz de confusões para o modelo treinado por uma época, em cima do conjunto de teste. A principal confusão que ocorre são textos de "Education & Reference" classificados como "Science & Mathematics".

A.3.1.2 Avaliação no conjunto de validação

A Tabela 13 mostra os resultados do modelo de uma época sobre o conjunto de validação.

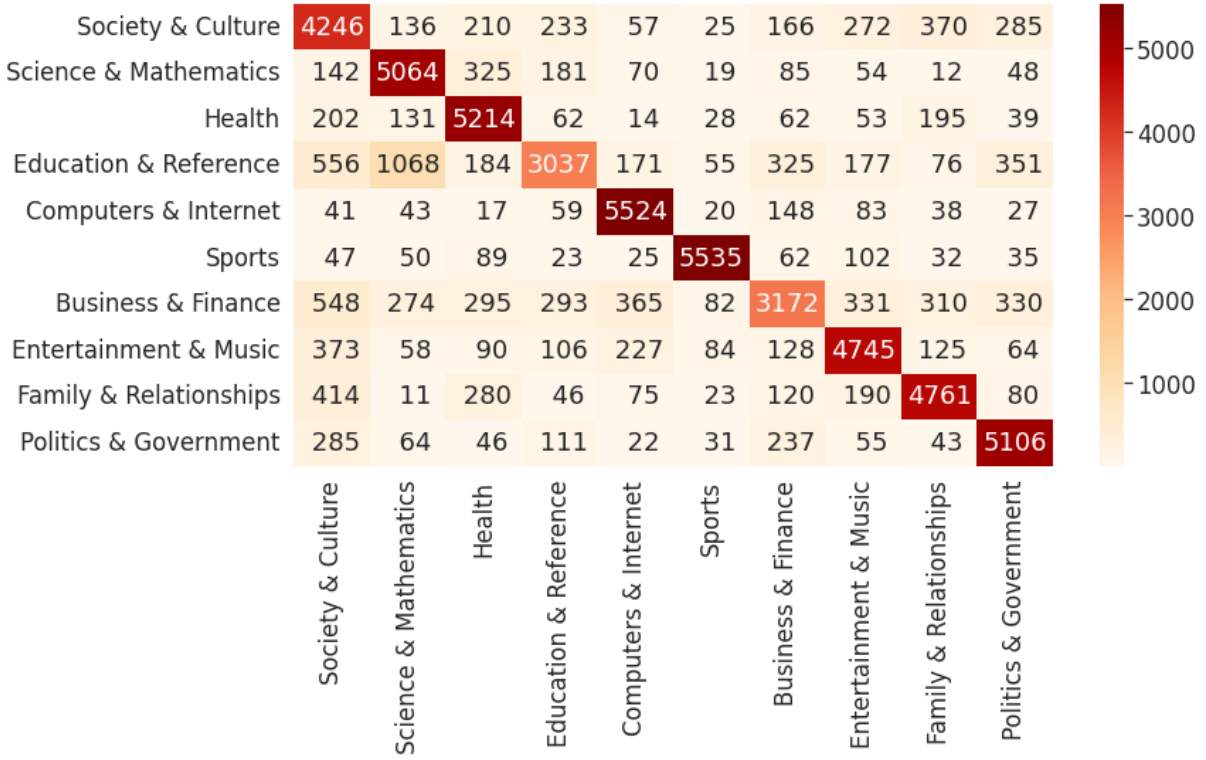


Figura 79 – Matriz de confusão do modelo treinado por 1 época, avaliado no conjunto de teste

Tabela 13 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de validação

Classe	Precisão	Recall	F1-Score
Society & Culture	62.32	70.47	66.14
Science & Mathematics	72.74	84.90	78.35
Health	77.99	87.14	82.31
Education & Reference	72.64	50.03	59.25
Computers & Internet	84.43	91.91	88.01
Sports	93.57	92.20	92.88
Business & Finance	69.52	52.91	60.09
Entertainment & Music	78.47	78.71	78.59
Family & Relationships	79.19	78.67	78.93
Politics & Government	79.53	84.63	82.00

A Figura 80 mostra as confusões no conjunto de validação, que seguem a mesma linha do conjunto de teste.

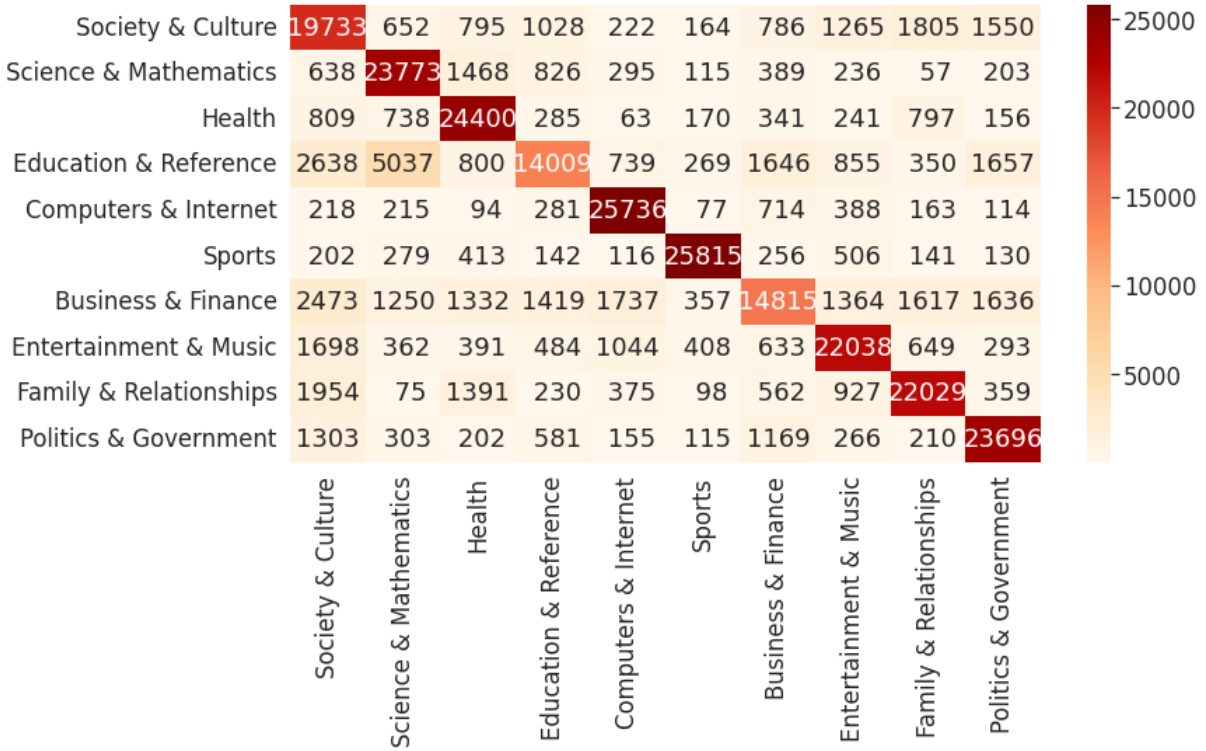


Figura 80 – Matriz de confusão do modelo treinado por 1 época, avaliado no conjunto de validação

A.3.2 Modelo treinado por 2 épocas

A.3.2.1 Avaliação no conjunto de teste

A Tabela 14 mostra os resultados do modelo treinado por duas épocas.

Tabela 14 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de teste

Classe	Precisão	Recall	F1-Score
Society & Culture	66.27	66.52	66.39
Science & Mathematics	75.69	82.42	78.91
Health	80.26	84.72	82.43
Education & Reference	66.62	58.75	62.44
Computers & Internet	86.71	90.98	88.79
Sports	92.82	92.67	92.74
Business & Finance	67.45	55.37	60.81
Entertainment & Music	78.42	79.22	78.82
Family & Relationships	77.56	81.82	79.63
Politics & Government	80.92	83.88	82.37

A Figura 81 mostra a matriz de confusão do modelo.

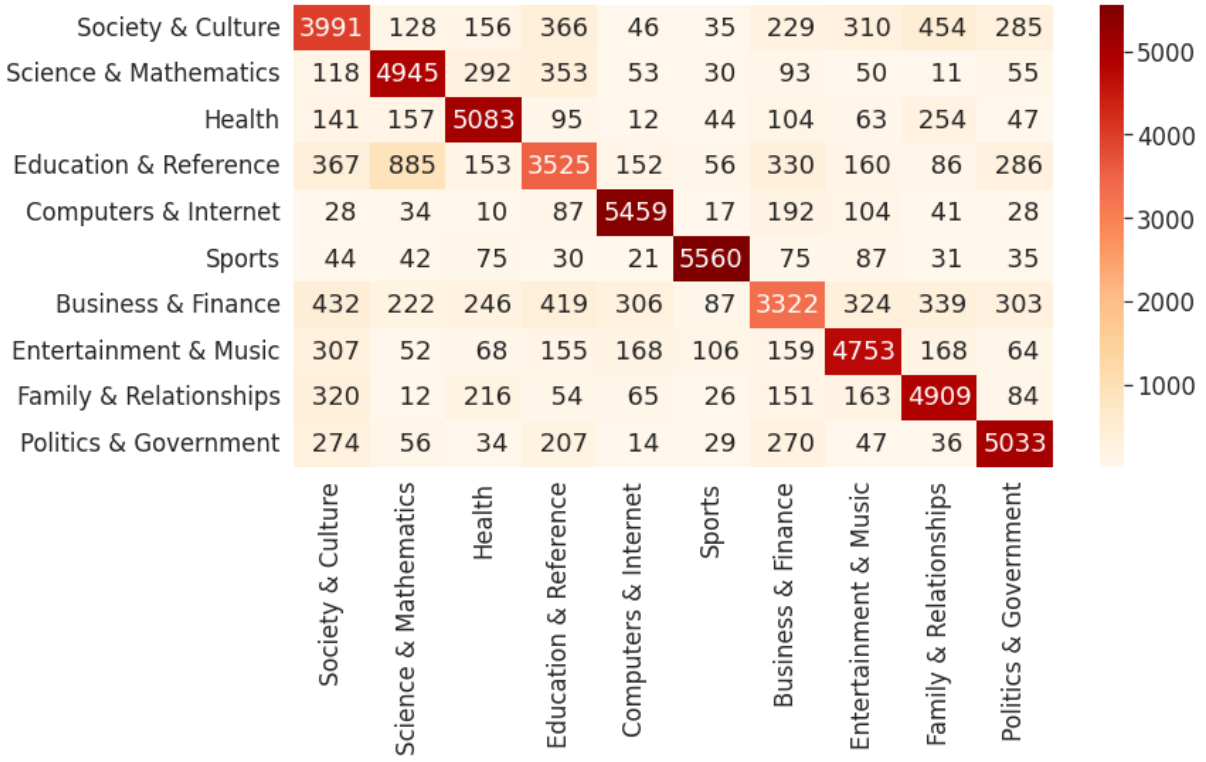


Figura 81 – Matriz de confusão do modelo treinado por 2 épocas, avaliado no conjunto de teste

A.3.2.2 Avaliação no conjunto de validação

A Tabela 15 mostra os resultados do modelo treinado por duas épocas, avaliado sobre o conjunto de validação.

Tabela 15 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de validação

Classe	Precisão	Recall	F1-Score
Society & Culture	66.49	66.60	66.54
Science & Mathematics	75.03	82.13	78.42
Health	80.80	85.04	82.86
Education & Reference	66.09	58.17	61.88
Computers & Internet	86.44	90.36	88.36
Sports	93.06	92.65	92.85
Business & Finance	66.41	55.38	60.39
Entertainment & Music	78.67	79.33	79.00
Family & Relationships	77.10	81.20	79.10
Politics & Government	80.63	83.43	82.01

A Figura 82 mostra a matriz de confusão do modelo.

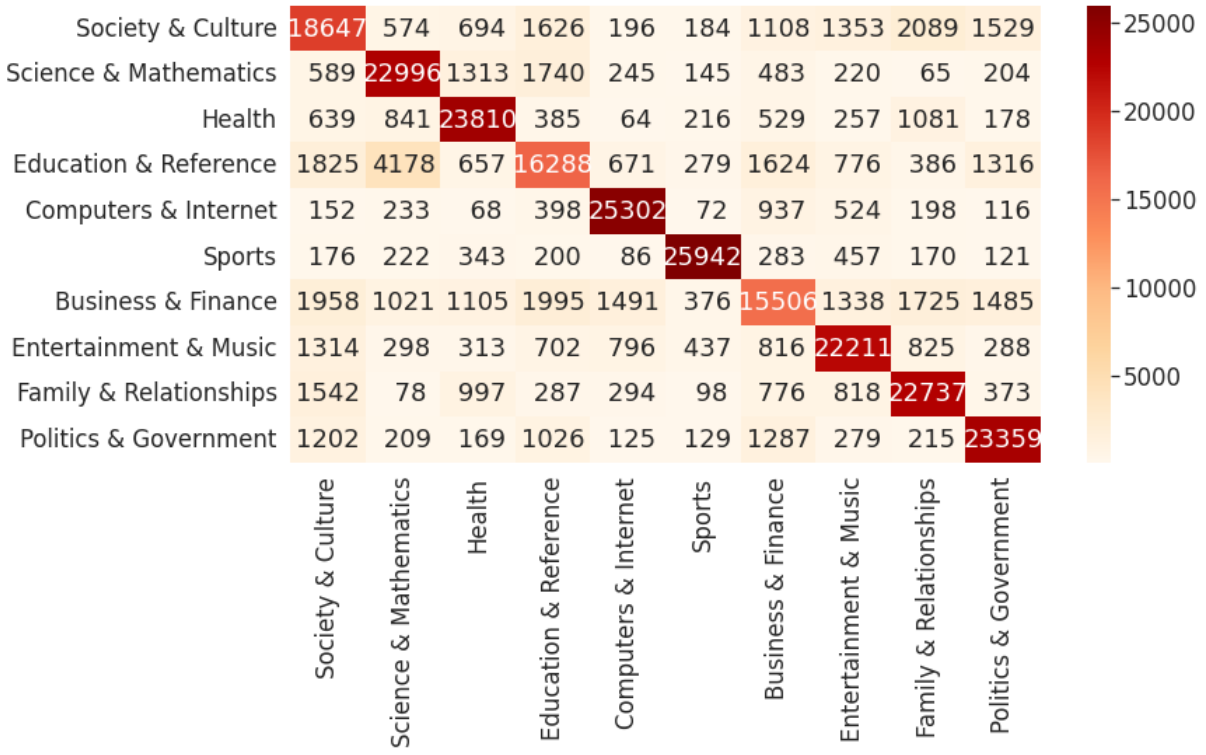


Figura 82 – Matriz de confusão do modelo treinado por 2 épocas, avaliado no conjunto de validação

APÊNDICE B – Modelo após mitigação dos vieses

B.1 Resultados obtidos

A Tabela 16 mostra o desempenho obtido pelo modelo conforme o conjunto de teste e o de validação, separado entre o modelo treinado por uma época e o modelo que passou por duas épocas de treinamento.

Tabela 16 – Métricas gerais: F1-Score obtido pelo BERT após 1 e também após 2 épocas de treinamento, avaliados nos conjuntos de teste e de validação, para o modelo treinado com o objetivo de mitigar os vieses

Conjunto	Épocas de Treinamento	F1-Score
Teste	1	76,70
Teste	2	76,69
Validação	1	76,59
Validação	2	76,52

A Figura 83 mostra a curva de aprendizado do modelo. O menor erro no conjunto de validação ocorre exatamente após uma época de treinamento. Nota-se o overfitting profundo alcançado pelo modelo da primeira época em diante, sem benefício no desempenho em relação ao conjunto de validação.

As subseções a seguir detalham o resultado de cada modelo por classe.

B.1.1 Modelo treinado por 1 época

B.1.1.1 Avaliação no conjunto de teste

Os resultados obtidos pelo modelo, após 1 época completa de treinamento, avaliados no conjunto de teste, estão na Tabela 17. A classe mais fácil para o modelo é "Sports", seguida por "Computer & Internet", enquanto as mais problemáticas são "Education & Reference" e "Business & Finance".

A Figura 84 apresenta a matriz de confusões para o modelo treinado por uma época, em cima do conjunto de teste, após a mitigação dos vieses. A principal confusão que ocorre são textos de "Education & Reference" classificados como "Science & Mathematics".

Figura 83 – Curva de aprendizado do BERT treinado, conforme a loss de entropia cruzada. Os valores de treinamento referem-se à loss média nos últimos 200 mil exemplos vistos, enquanto a validação é sempre feita no conjunto completo de validação

Tabela 17 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de teste, após a mitigação dos vieses

Classe	Precisão	Recall	F1-Score
Society & Culture	67.31	63.57	65.39
Science & Mathematics	75.66	81.67	78.55
Health	78.41	84.92	81.53
Education & Reference	66.21	55.95	60.65
Computers & Internet	88.18	88.75	88.46
Sports	93.27	92.17	92.72
Business & Finance	67.92	54.88	60.71
Entertainment & Music	78.22	79.18	78.70
Family & Relationships	72.03	85.28	78.10
Politics & Government	80.29	84.25	82.22

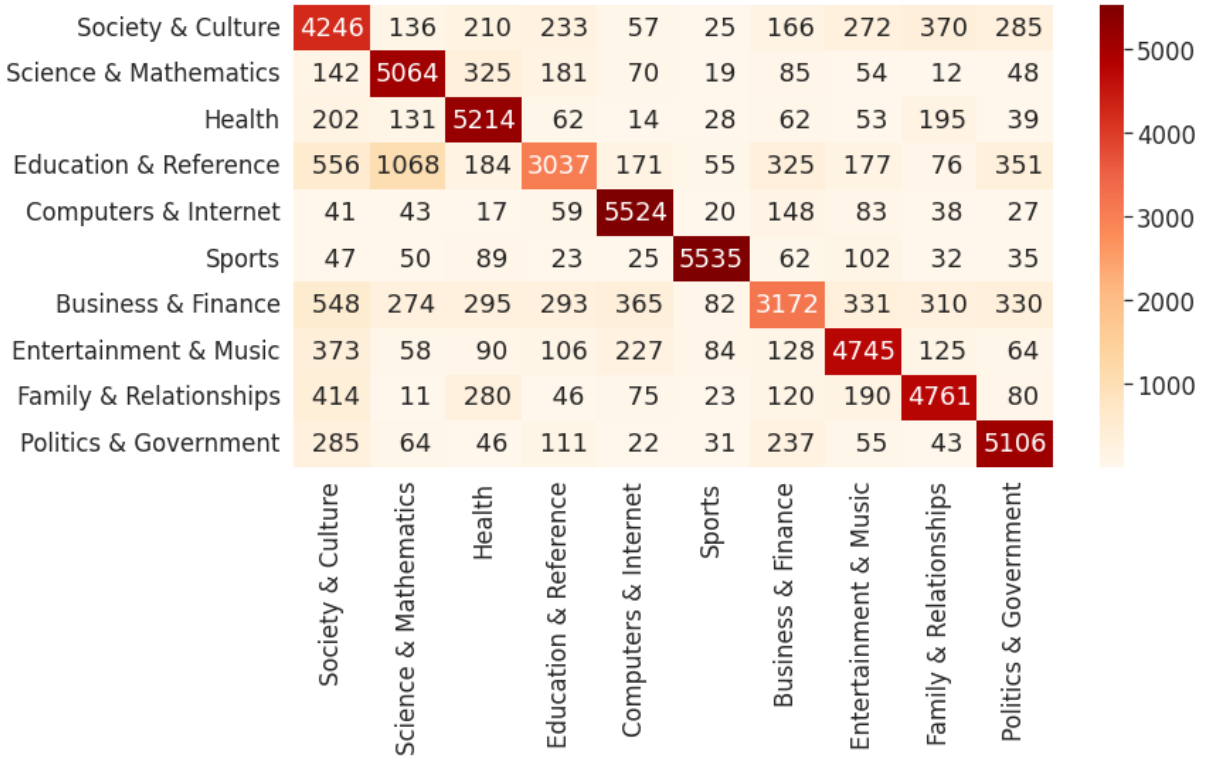


Figura 84 – Matriz de confusão do modelo treinado por 1 época, avaliado no conjunto de teste, após a mitigação dos vieses

B.1.1.2 Avaliação no conjunto de validação

A Tabela 18 mostra os resultados do modelo de uma época sobre o conjunto de validação, após a mitigação dos vieses.

Tabela 18 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de validação, após a mitigação dos vieses

Classe	Precisão	Recall	F1-Score
Society & Culture	67.21	63.91	65.52
Science & Mathematics	75.17	81.88	78.38
Health	78.68	84.81	81.63
Education & Reference	66.44	55.27	60.34
Computers & Internet	88.25	88.70	88.47
Sports	93.27	92.05	92.66
Business & Finance	67.06	55.05	60.46
Entertainment & Music	79.08	78.81	78.94
Family & Relationships	71.69	85.02	77.79
Politics & Government	79.59	83.95	81.71

A Figura 85 mostra as confusões no conjunto de validação, que seguem a mesma linha do conjunto de teste.

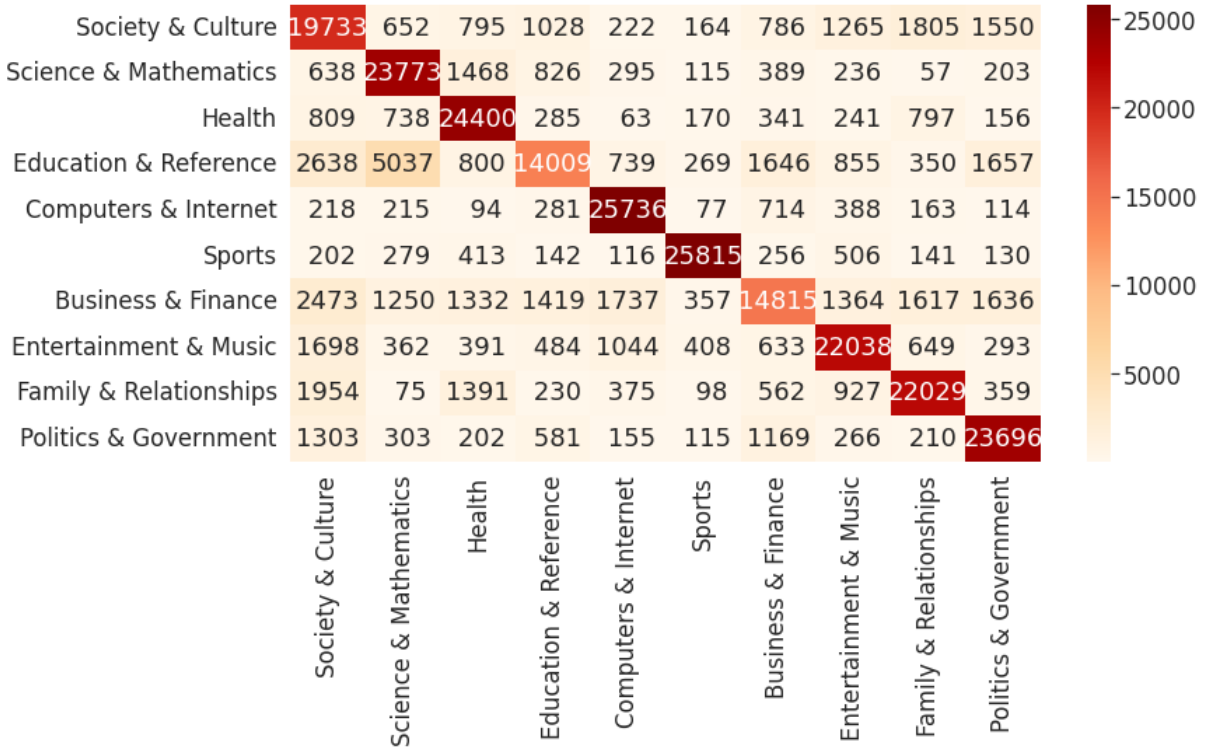


Figura 85 – Matriz de confusão do modelo treinado por 1 época, avaliado no conjunto de validação, após a mitigação dos vieses

B.1.2 Modelo treinado por 2 épocas

B.1.2.1 Avaliação no conjunto de teste

A Tabela 19 mostra os resultados do modelo treinado por duas épocas com o dataset preparado para a mitigação dos vieses.

Tabela 19 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de teste

Classe	Precisão	Recall	F1-Score
Society & Culture	66.14	65.97	66.05
Science & Mathematics	73.71	83.28	78.21
Health	80.49	82.58	81.52
Education & Reference	66.51	53.95	59.57
Computers & Internet	86.56	90.27	88.37
Sports	92.94	92.40	92.67
Business & Finance	67.04	55.22	60.56
Entertainment & Music	77.61	79.53	78.56
Family & Relationships	75.90	82.95	79.27
Politics & Government	79.89	84.47	82.11

A Figura 86 mostra a matriz de confusão do modelo.

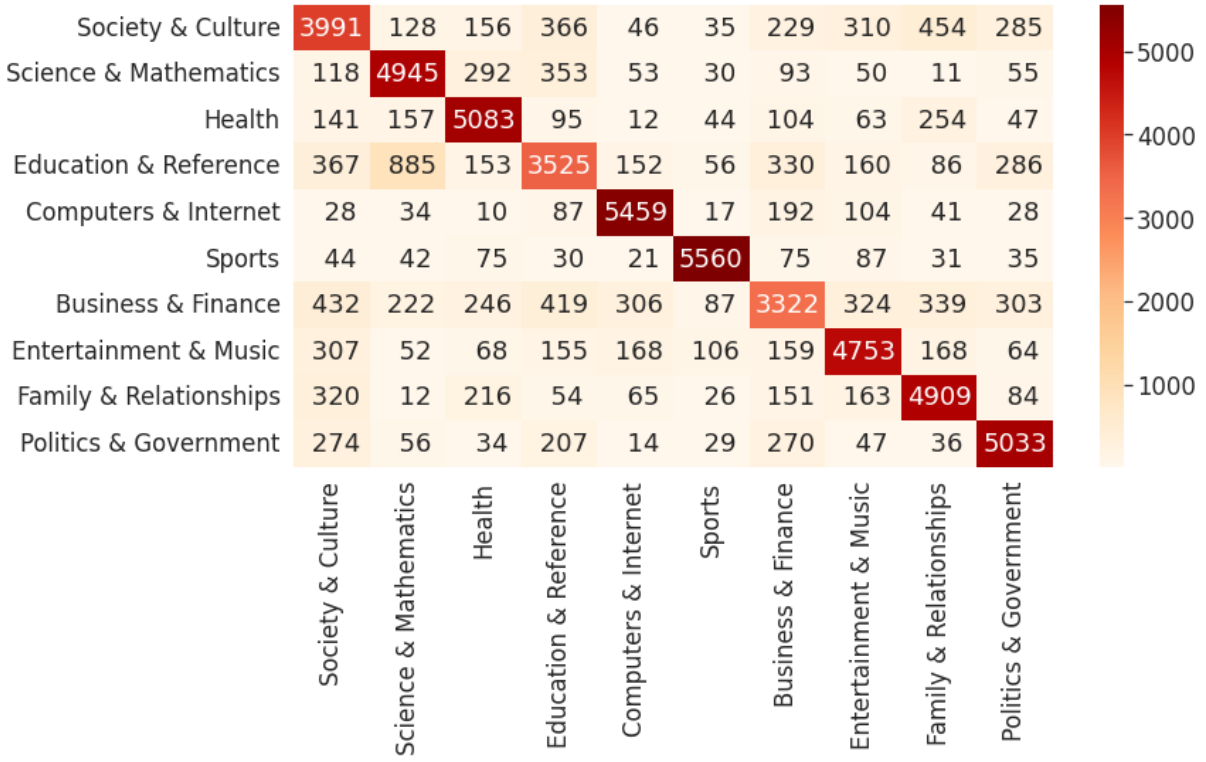


Figura 86 – Matriz de confusão do modelo treinado por 2 épocas, avaliado no conjunto de teste

B.1.2.2 Avaliação no conjunto de validação

A Tabela 20 mostra os resultados do modelo treinado por duas épocas, avaliado sobre o conjunto de validação.

Tabela 20 – Precisão, recall e f1-score de cada classe, conforme avaliado no conjunto de validação

Classe	Precisão	Recall	F1-Score
Society & Culture	65.79	65.75	65.77
Science & Mathematics	73.58	83.34	78.16
Health	81.14	82.90	82.01
Education & Reference	66.32	53.53	59.24
Computers & Internet	86.76	89.89	88.30
Sports	92.89	92.31	92.60
Business & Finance	65.86	55.27	60.10
Entertainment & Music	78.30	79.35	78.82
Family & Relationships	75.13	82.02	78.42
Politics & Government	79.24	84.40	81.74

A Figura 87 mostra a matriz de confusão do modelo.

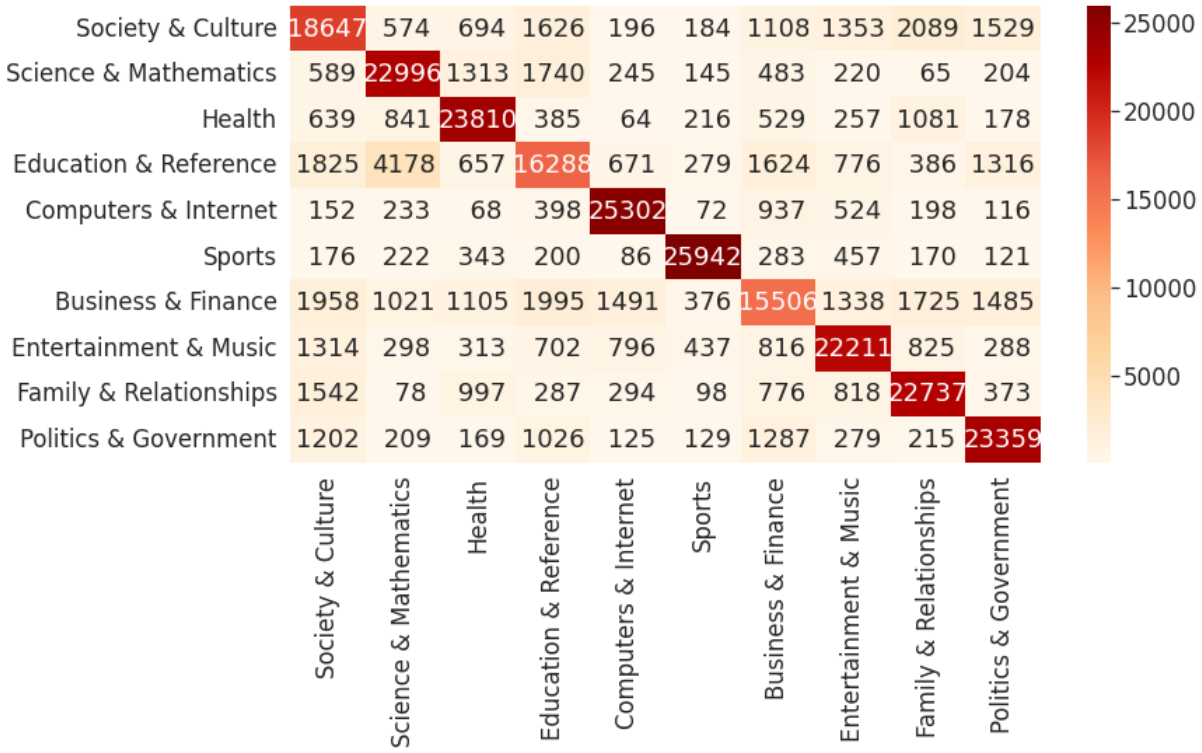


Figura 87 – Matriz de confusão do modelo treinado por 2 épocas, avaliado no conjunto de validação

APÊNDICE C – Taxas de falsos positivos e falsos negativos após a mitigação dos vieses identificados

C.1 Resultados

A Tabela 21 mostra as taxas de falsos positivos e falsos negativos para o modelo treinado com o dataset para mitigação dos vieses identificados.

Tabela 21 – Taxa de falsos positivos e falsos negativos para cada par termo-classe, para o modelo treinado com o dataset para mitigação dos viéses

Termo	Classe	FPR (1 época)	FPR (2 épocas)	FNR (1 época)	FNR (2 épocas)
back	Health	2,11%	1,78%	10,64%	12,77%
thigh	Health	33,33%	33,33%	0%	0%
medicine	Health	14,06%	6,25%	17,97%	39,84%
data	Computer & Internet	5,76%	5,76%	4,58%	9,92%
baseball	Sports	8,33%	6,25%	9,27%	13,31%
802 (802.11)	Computers & Internet	28,75%	28,57%	0%	0%
teams	Sports	17,07%	12,20%	0,94%	2,19%
community	Education & Reference	7,44%	5,79%	28,09%	34,83%
eddie	Entertainment & Music	6,45%	6,45%	0%	0%
school	Education & Reference	7,37%	5,07%	27,39%	37,92%
health	Health	8,12%	6,25%	21,70%	28,18%
marriage	Family & Relationships	8,82%	4,12%	16,67%	26,39%
gay	Society & Culture	7,56%	9,28%	47,37%	46,84%
homework	Education & Reference	9,79%	6,99%	39,01%	46,81%