Pós-Graduação em Ciência da Computação

DAVID WILSON DE FARIAS SANTOS

Predição de popularidade em mídia social utilizando uma rede de atenção hierárquica

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
http://cin.ufpe.br/~posgraduacao

Recife
2019

DAVID WILSON DE FARIAS SANTOS

Predição de popularidade em mídia social utilizando uma rede de atenção hierárquica

Dissertação apresentada ao Programa de Pós-Graduação em Ciências da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciências da Computação.

Área de Concentração: Inteligência Computacional.

Orientador: Prof. Dr. Tsang Ing Ren

Recife
2019

Catalogação na fonte
Bibliotecária Mariana de Souza Alves CRB4-2106

S237p Santos, David Wilson de Farias
Predição de popularidade em mídia social utilizando uma rede de atenção hierárquica – 2019.
101f.: il., fig., tab.

Orientador: Tsang Ing Ren
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIN, Ciência da Computação. Recife, 2019.
Inclui referências.

1. Inteligência Computacional. 2. Mídia Social. 3. Redes de Atenção. 4. Dados Visuais. I. Ren, Tsang Ing (orientador). II. Título.

006.31 CDD (22. ed.) UFPE-MEI 2019-139

David Wilson de Farias Santos

“Predição de popularidade em mídia social utilizando rede de atenção hierárquica”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 23 de agosto de 2019.

BANCA EXAMINADORA

_____

Prof. Dr. Luciano de Andrade Barbosa
Centro de Informática / UFPE

_____

Prof. Dr. Renato Vimieiro
Departamento de Ciência da Computação/UFMG

_____

Prof. Dr. Tsang Ing Ren
Centro de Informática / UFPE
(**Orientador**)

AGRADECIMENTOS

Agradeço a Deus por me conceder saúde, determinação e inspiração para superar todas as dificuldades. Aos meus pais, Carme Célia e Wilson José, que compreenderam minha ausência durante esses últimos meses e que me deram suporte e educação suficiente para chegar até aqui. A minha esposa, Marília Cavalcanti, pela compreensão, paciência e dedicação durante todo esse tempo em que a prioridade era sempre "o mestrado".

Aos meus sogros pelo apoio nos momentos necessários. Aos meus amigos e colegas de trabalho, Walber Rodrigues e João Nunes, pelas trocas de ideias e conhecimentos que contribuíram para o término deste trabalho; e a Heitor Araújo por todas as caronas e conselhos concedidos durante esse período.

Aos responsáveis e colegas do ISI-Tics, principalmente Sérgio Soares e Bruno Medeiros, pela compreensão em momentos que tive que me ausentar e trabalhar de casa. Agradeço ao meu orientador Tsang Ing Ren por toda confiança, por todas as reuniões/orientações e ensinamentos que foram importantes para a construção deste trabalho.

Agradeço especialmente a plataforma em nuvem do Google por disponibilizar um ambiente que possibilitou a realização de todos os meus experimentos de forma rápida e eficaz. Por fim, a todos que de alguma forma contribuíram e me motivaram durante esses dois anos de caminhada.

RESUMO

Diariamente, milhares de conteúdos textuais e visuais são compartilhadas entre as pessoas nas redes sociais, entretanto, enquanto alguns conteúdos conseguem engajar vários usuários, outros são totalmente ignorados não alcançando a popularidade que é tão almejada nessas plataformas. Devido a isso, houve um crescimento do interesse dos cientistas de dados na tentativa de entender o que torna alguns conteúdos mais relevantes do que outros. Pois, a compreensão do que torna um conteúdo popular é uma oportunidade sem precedentes para grandes aplicações comerciais, tais como sistemas de recomendação e *marketing* digital que quer aproveitar ao máximo a atenção do público. Por essas razões o problema de predição de popularidade em mídias sociais tem sido estudado extensivamente ao longo dos últimos anos. Pesquisas anteriores seguem abordagens que atuam sobre dados textuais, visuais e características sociais do perfil. Apesar de alcançarem resultados promissores, os modelos obtidos apresentam pouca capacidade de generalização, devido ao uso de dados direcionados a usuários específicos, dificultando assim a predição de popularidade de novos conteúdos. Portanto, será proposta uma Rede de Atenção Hierárquica Estendida (RAHE) que atende hierarquicamente dados textuais e visuais, levando em conta a influência de um contexto compartilhado entre os usuários de características sociais em comum, representações textuais a nível de caracteres e um método que minimiza os erros obtidos pela predição de popularidade. O objetivo dessa abordagem é tratar os problemas de diversidade e capacidade de generalização encontradas por grande parte dos modelos presentes no estado da arte no problema de predição de popularidade de novos conteúdos. Os experimentos realizados sobre a base de dados *Temporal Popularity Image Collection* (TPIC) do Flickr demostraram que o modelo RAHE, além de tratar bem a diversidade, apresenta uma alta capacidade de generalização, pois, consegue reduzir significativamente o erro médio quadrático obtido pelo *baseline* na predição de popularidade de imagens publicadas por novos usuários em 48,14%.

Palavras-chaves: Mídia Social. Redes de Atenção. Dados Visuais. Dados Textuais a Nível de Caractere. Minimização de Erro. Predição de Popularidade.

ABSTRACT

Every day thousands of textual and visual information are shared among people every minute, however some content attracts the attention of many users; while others are totally ignored, not reaching the popularity that is so desired on these platforms. Because of this, there has been a growing interest from scientists in an attempt to understand what makes some content more relevant than others. Because understanding what makes content popular is an unprecedented opportunity for large commercial applications like recommendation systems and digital marketing that want to get the most out of the public's attention. For these reasons, the problem of predicting popularity on social media has been studied extensively in recent years. Previous research follows approaches that act on textual, visual and social characteristics of the profile.Although achieving promising results, the models achieved have little generalization capacity due to the use of specific users data, thus making it difficult to predict new content. Therefore, it will be proposed an Extended Hierarchical Attention Network (RAHE) which hierarchically attends textual and visual data, taking into account the influence of a shared context among users of common social characteristics, textual representations at character-level and a method which minimizes error obtained by popularity predicting. This approach aims to address the diversity and generalizability problems encountered by most state-of-the-art models in the popularity predicting problem of new content. The experiments performed on Flickr's Temporal Popularity Image Collection (TPIC) database showed that the RAHE model, besides handling diversity as well, has a high generalization capacity, since it can significantly reduce the mean square error obtained by the popularity prediction of images posted by new users at 48.14%.

Keywords: Social Media. Attention Network. Visual Data. Error Correction. Popularity Prediction. Textual Representations at Character-Level.

LISTA DE FIGURAS

LISTA DE TABELAS

SIGLAS

ANPs	*Adjective-Noun-Pairs.*
API	Application Programming Interface.
CNN	Redes Neurais Convolucionais.
DANs	*Redes de Atenção Dupla.*
DNN	Redes Neurais Profundas.
DTCN	Redes Temporais de Contexto Profundo.
ILSVRC	*ImageNet Large-Scale Visual Recognition Challenge.*
LSTM	*Long Short Term Memory.*
MAE	Erro Médio Absoluto.
MSE	Erro Médio Quadrático.
NLP	Processamento de Linguagem Natural.
RAHE	Rede de Atenção Hierárquica Estendida.
RAHE-CE	Rede de Atenção Hierárquica Estendida com Correção de Erro.
ReLU	*Rectified Linear Unit.*
RNN	Rede Neural Recorrente.
TPIC	*Temporal Popularity Image Collection.*
UHAN	Rede de Atenção Hierárquica guiada por Usuário.
VQA	*Visual Question Answer*.

SUMÁRIO

1 INTRODUÇÃO

A Web 2.0 trouxe uma maneira de incluir a participação e a colaboração contínua dos usuários no conteúdo gerado na internet, tornando-os capazes de compartilhar informações e experiências uns com os outros todos os dias através das mídias sociais. Esses dados compartilhados acabam por descrever interesses pessoais, atividades diárias, opiniões e o que está acontecendo a sua volta. Geralmente, o compartilhamento de conteúdo e interação com outros usuários é motivada pelo interesse de alcançar uma presença social ativa ou um certo nível de popularidade (NOV; NAAMAN; YE, 2008).

Entretanto, apenas alguns conteúdos tem a capacidade de atrair a atenção de outros usuários, enquanto muitos outros passam despercebidos. Até os conteúdos publicados por um mesmo usuário podem apresentar variações, onde, em algumas ocasiões, conseguem despertar o interesse de um certo público e, em outras, apresentam um tipo de conteúdo que simplesmente é ignorado.

Nos últimos anos, a previsão da popularidade de publicações geradas pelos usuários nas redes sociais tem chamado a atenção não só dos pesquisadores, como também de alguns setores da indústria. Pois, entender a dinâmica desses dados sociais e o comportamento dos usuários é uma oportunidade sem precedentes para grandes aplicações comerciais, tais como sistemas de recomendação de conteúdo (KHOSLA; SARMA; HAMID, 2014) e *marketing digital* (MAZLOOM et al., 2016).

A predição da popularidade tenta inferir a quantidade de interações entre os usuários pertencentes a uma determinada mídia social e o conteúdo gerado por outro usuário. Esses tipos de interações podem variar de acordo com o tipo de mídia social, pois, cada uma apresenta um tipo de medida específica, podendo ser representada por cliques, visualizações, *likes*, compartilhamentos, etc. Além disso, o tipo de conteúdo também varia de uma plataforma para outra, sendo os mais comuns entre as plataformas: o conteúdo visual, que se trata da imagem utilizada na publicação; o conteúdo textual, que pode estar associado ou não a uma imagem; e o conteúdo do usuário, que apresenta informações sociais sobre o perfil do usuário.

Então, a partir dos dados visuais, textuais e do usuário, como é possível obter a popularidade de um novo conteúdo gerado pelo usuário? Quais características devem ser exploradas de cada um dos dados presentes no conteúdo gerado pelo usuário?

Em problemas de predição de popularidade diferentes características do perfil do usuário e do conteúdo gerado são extraídas e analisadas. Aloufi, Zhu e El Saddik (2017), Khosla, Sarma e Hamid (2014) e McParlane, Moshfeghi e Jose (2014) apresentaram resultados que demostraram a influência positiva das características sociais do perfil (Ex.: quantidade de seguidores, grupos, média de visualizações e etc.) para a predição de popularidade. Esses mesmos estudos também avaliaram, que, apesar da imagem ser um dos principais recursos

utilizados nas pesquisas, não apresenta a mesma influência que as características sociais do perfil.

Nos estudos realizados por Gelli et al. (2015) e Wu et al. (2017) foram exploradas, respectivamente: características visuais de sentimento, com objetivo de entender a relação entre o sentimento e a popularidade; e informações temporais, como semana do ano e período do dia, para entender a influência que dados temporais apresentam sobre a popularidade.

Wang, Sun e Chen (2019) e Zhang et al. (2018) utilizaram multimodalidades em conjunto com as técnicas de mecanismos de atenção, devido ao sucesso obtido por essa combinação aplicada a outros problemas. Essas abordagens demostraram que modalidades visuais, textuais e de usuário aprendidas durante a etapa de treinamento e combinadas com os mecanismos de atenção apresentam performances superiores em comparação com os métodos que lidam com modalidades que não são aprendidas durante a etapa de treinamento.

Entretanto, muitos dos estudos realizados negligenciam o problema de popularidade em um desses dois aspectos: diversidade e capacidade de generalização. Wang, Sun e Chen (2019) demostrou que modelos que utilizam características sociais do perfil negligenciam a diversidade dos usuários, não refletindo corretamente a popularidade de um usuário específico. Em contrapartida, modelos que focam em usuários específicos acabam apresentando pouca capacidade de generalização.

Um outro ponto bastante relevante levantado por Aloufi, Zhu e El Saddik (2017), é que informação textual, não utilizada em muitas outras abordagens, não deve ser negligenciada ao prever a popularidade de uma imagem. Porém, a maioria das abordagens que utilizam a representação textual focam sempre na extração de características através do *tf-idf* e da LSTM a nível de palavras, não explorando a contribuição que a aplicação de outras técnicas sobre informações textuais a nível de caractere podem agregar ao problema de popularidade.

Além disso, também foi observado que a medida de popularidade, na maioria dos casos, segue uma distribuição gaussiana. Dessa forma, grande parte dos eventos e amostras se concentram em torno de um valor médio. Entretanto, uma outra parcela, representando amostras de baixa frequência no conjunto amostral, se concentra mais afastada desses valores médios. Nesse caso, por se tratarem de eventos de menor ocorrência, em relação aos dados que estão em torno da média, são mais difíceis de serem previstos, obtendo assim erros mais elevados.

Portanto, o uso de modelos com multimodalidades obtidas a partir de técnicas de aprendizagem profunda, ainda não foi explorado para tratar os problemas de diversidade e capacidade de generalização para predição de popularidade levando em conta a incorporação de novas características textuais e a minimização dos erros obtidos para amostras de baixa frequência no conjunto amostral de popularidade.

1.1 OBJETIVOS

Esse estudo tem como objetivo apresentar um modelo que utiliza modalidades visuais, textuais e do usuário, para tratar os problemas de diversidade e capacidade de generalização, e minimizar os erros obtidos para amostras de baixa frequência no conjunto amostral. Para atender esse objetivo, o estudo apresenta os seguintes objetivos específicos:

- Criar uma nova representação do usuário que leva em consideração um contexto genérico compartilhado entre usuários com caraterísticas sociais similares.
- Avaliar uma representação textual que incorpore novas características textuais, explorando a representação textual a nível de caractere com intuito de entender a contribuição textual de *n-grams* a nível de caractere para o problema de popularidade.
- Propor um modelo de Rede de Atenção Hierárquica Estendida (RAHE) que atende hierarquicamente o contexto compartilhado entre usuários e representações textuais que incorporem características textuais a nível de caractere, para tratar os problemas de diversidade e capacidade de generalização em problemas de predição de popularidade.
- Propor um método que tem como propósito aprender e minimizar os erros obtidos pela predição de popularidade do modelo RAHE, principalmente, os erros relacionados as amostras de baixa frequência no conjunto amostral.

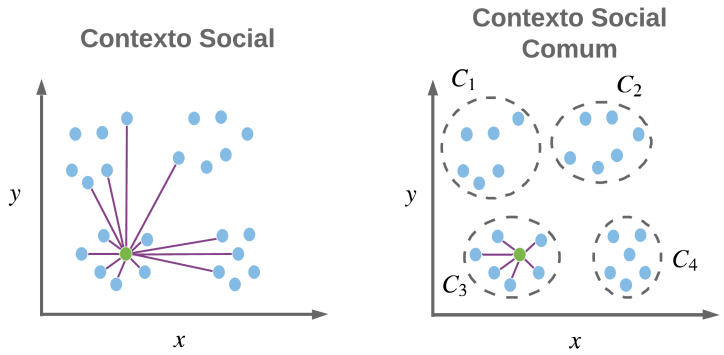


Figura 1 – Exemplo do uso do contexto social e do contexto social comum: Em relação ao contexto social, ao chegar um novo conteúdo, o mesmo é confrontado com informações de contextos externos que estão fora do seu círculo imediato, o que pode confundir na tomada de decisão; no contexto social comum, o novo conteúdo é confrontado apenas com as informações e experiências presentes no seu contexto, não sofrendo interferência de informações de contextos externos na sua tomada de decisão.

A intuição por trás do contexto genérico pode ser demostrada pela Figura 1, em que usuários que apresentam características similares auxiliam uns aos outros na predição da popularidade de novos conteúdos, compartilhando informações e comportamentos já decorridos. Onde, essas informações e comportamentos são representados por informações visuais e textuais de cada usuário pertencente a um mesmo contexto.

Esse tipo de abordagem é útil, pois, em alguns casos em que um usuário publica um conteúdo que não segue os padrões normais do seu perfil, é possível, a partir de informações de outros usuários presentes no contexto em que esse usuário está imerso, saber se aquele tipo de conteúdo pode ser popular ou não.

1.2 CONTRIBUIÇÕES

Uma das contribuições dessa pesquisa se trata da base de dados gerada, pois, foi despendido um tempo considerável para obtê-la. Devido a pesquisa lidar com dados não estruturados, como texto e imagem, é necessário ter ao menos uma base com essas informações. Entretanto, todas as bases disponíveis que tratem o problema de popularidade não os apresentam.

Então, foi construída uma base de dados, com tais representações, estendendo a base de dados pública *Temporal Popularity Image Collection* (TPIC), na qual, será disponibilizada em um repositório para facilitar o desenvolvimento de trabalhos de outros pesquisadores que se interessem pela área de previsão de popularidade. Dessa forma, o modelo RAHE foi testado sobre conteúdos providos da mídia social Flickr[1]

A partir dessa base de dados algumas análises foram realizadas e foi observado que o modelo RAHE apresenta um desempenho que supera os resultados dos demais modelos existentes no estado da arte em todos os cenários, cobrindo os aspectos de diversidade e capacidade de generalização, tendo assim como principais contribuições dessa pesquisa:

- Definição de cenários que realizam a cobertura da diversidade e capacidade de generalização dos modelos. Apesar de algumas abordagens na literatura testarem seus modelos em alguns cenários, esses cenários são insuficientes para cobrir a questão da diversidade e a capacidade de generalização dos modelos.

- A concepção de um contexto compartilhado entre os usuários de características sociais em comum, que se trata de uma nova representação das características sociais do perfil do usuário. Dessa forma, é possível que perfis compartilhem experiências e comportamentos entre si a partir de informações textuais e visuais extraídas pelo modelo.

[1] Um site da web de hospedagem e partilha de imagens como fotografias, desenhos e ilustrações, além de permitir novas maneiras de organizar as fotos e vídeos. Caracterizado como rede social, o site permite aos usuários criar álbuns para armazenar suas fotografias e contatar-se com usuários de diferentes locais do mundo.

- A partir do *embedding* gerado para contexto compartilhado entre os usuários de características sociais em comum, foi possível visualizar a geração de grupos específicos e evidenciar que usuários com contextos similares estão próximos no espaço contextual e estão correlacionados com o nível de popularidade.
- A integração de um método de correção de erro que tem como propósito minimizar os erros obtidos pela predição de popularidade, principalmente, os erros relacionados as amostras de baixa frequência no conjunto amostral, utilizando apenas uma rede neural *feedfoward* de duas camadas.
- Análise da contribuição de informações textuais a nível de caractere, tendo a CNN como unidade extratora de características a nível de caracteres, para lidar com o problema de predição de popularidade.

1.3 ESTRUTURA DA DISSERTAÇÃO

A dissertação está estruturada no seguinte formato: O Capítulo 2 apresenta o estado da arte na predição de popularidade. Capitulo 3 apresenta toda fundamentação teórica, que serviu de base para o desenvolvimento dessa pesquisa. No Capítulo 4 é explicada a estrutura da RAHE, abordando tópicos de como são obtidas as representações visuais, o passo a passo de como são obtidas as representações textuais a nível de palavra e caractere, como foram definidas as representações do usuário, como as camadas de atenção hierárquica extraem as multimodalidades e como o módulo de correção de erro é integrado ao modelo RAHE. No Capítulo 5 são fornecidos detalhes de como foram conduzidos os experimentos, onde é dada uma visão geral de como foi obtida a base de dados, os processamentos realizados sobre a base para obter as informações textuais, visuais e do usuário, como foi realizada a divisão dos dados da base de dados, as métricas de avalição dos modelos, os *baselines* que serão comparados com a abordagem desenvolvida e o ambiente de desenvolvimento. O Capítulo 6 apresenta todos os estudos realizados, como a contribuição individual das características sociais, visuais e textuais, o desempenho da combinação entre modalidades, a comparação do modelo RAHE como os demais modelos no estado da arte, a relevância das representações textuais para tratar o problema de popularidade, o desempenho do modelo com o módulo de correção de erro integrado, as informações que o *embedding* contextual pode fornecer e o comportamento do erro médio obtida pelo método de correção de erro. No Capítulo 7 contém as conclusões acerca da pesquisa desenvolvida e trabalhos futuros.

2 TRABALHOS RELACIONADOS

Essa seção apresenta uma revisão do estado da arte em predição de popularidade que serviram de motivação na elaboração da proposta que contempla a diversidade e a capacidade de generalização dos modelos.

2.1 PREDIÇÃO DE POPULARIDADE

Diariamente, centenas de milhares de fotografias são transferidas para internet a cada minuto através de várias redes sociais e plataformas de compartilhamento de fotos. Enquanto algumas imagens obtêm milhões de visualizações, outras são completamente ignoradas. Ainda que o conteúdo seja publicado pelo mesmo usuário, existirá diferentes alcances para cada conteúdo. Entender a dinâmica desses dados sociais e o comportamento dos usuários pode ajudar na análise e interpretação de vários sistemas e no fornecimento serviços para grandes aplicações comerciais, tais como, sistemas de recomendação (KHOSLA; SARMA; HAMID, 2014) e *marketing* digital (MAZLOOM; HENDRIKS; WORRING, 2017).

Por essas razões, o problema de predição de popularidade das publicações geradas pelos usuários nas redes sociais tem sido estudado extensivamente ao longo dos últimos anos por causa de suas aplicações difundidas. Khosla, Sarma e Hamid (2014), Aloufi, Zhu e El Saddik (2017) e McParlane, Moshfeghi e Jose (2014) avaliaram a influência das características do sociais do perfil do usuário (Ex.: quantidade de seguidores, grupos, média de visualizações e etc.) na predição de popularidade. Entretanto, Wang, Sun e Chen (2019) demostrou que modelos que utilizam contexto social negligenciam as características de diversidade dos usuários, não refletindo corretamente a popularidade de um usuário particular.

Esses mesmos estudos também avaliaram, que, apesar da imagem ser um dos principais recursos utilizados nas pesquisas, não apresentam a mesma influência que as características sociais do perfil. Apesar de aplicar as mais variadas técnicas de extração de sobre as imagens (Ex.: *transfer learning* (SZEGEDY et al., 2017)), esses estudos não analisam a complementariedade da informação visual entre diferentes características para tratar o problema de predição de popularidade.

Aloufi, Zhu e El Saddik (2017) demostraram a importância do uso de representações textuais, que é negligenciada pela maioria das abordagens, porém, não só Aloufi, Zhu e El Saddik (2017), como outros estudos, utilizam extração manual de características da representação textual, como o tf-idf [1], não explorando a extração de características durante a etapa de treinamento, utilizando a Rede Neural Convolucional (CNN) (LECUN et al., 1989) e Rede Neural Recorrente (RNN) (ELMAN, 1990).

[1] *term frequency-inverse document frequency* é uma medida estatística usada para avaliar a importância de uma palavra para um documento em uma coleção ou corpus.

Alguns estudos começaram a extrair outras características dos conteúdos gerados pelos usuários, como por exemplo, o trabalho apresentado por Gelli et al. (2015) explorou o estado da arte na extração de características visuais de sentimento, a partir de *Adjective-Noun-Pairs* (ANPs) (BORTH et al., 2013), para entender a influência que sentimentos negativos e positivos apresentam na predição de popularidade. A partir dos estudos individuais realizados, foi demostrado qualitativamente, que as ANPs obtidas apresentam uma correlação com a popularidade.

Wu et al. (2016) propôs um novo modelo, nomeado de Decomposição Temporal em Várias Escalas (MTD), utiliza a informação temporal (Ex.: mês, dia, hora e etc.) dividida em múltiplas escalas e o contexto do item do usuário que compartilha comportamentos em comum dos usuários. Os resultados experimentais demonstraram a eficácia da decomposição em escala múltipla para predição de popularidade, onde duas escalas de tempo (semana do ano e período do dia) têm uma maior influência na predição de popularidade. Apesar dessas características extraidas por Gelli et al. (2015) e Wu et al. (2016) serem correlacionadas com a popularidade, essas correlações ainda são menores quando comparado com a obtida pelas características sociais do usuário.

Algumas abordagens começaram a investir tempo e esforço em modelos com multimodalidades (WELLS et al., 1996). Mazloom, Hendriks e Worring (2017) propôs uma técnica que leva em consideração as preferências dos usuários individuais para os itens no treinamento de um modelo de popularidade. Utiliza um contexto que combina informações do usuário, o interesse do usuário, o item e o contexto visual e textual, que é altamente correlacionado com o criador do conteúdo. Essa proposta, tem a capacidade de prever a popularidade de postagens relacionadas a itens diferentes que são compartilhados por um usuário específico. Além disso, pode prever a popularidade de postagens compartilhadas com usuários diferentes para um item específico. Apesar de apresentar eficácia na predição de popularidade, esse método necessita de informações de um produto ou marca utilizada por um usuário em sua publicação, impossibilitando a predição de popularidade sobre conteúdos que não apresentam essa informação.

O uso de multimodalidades com mecanismo de atenção foi utilizado nos mais recentes estudos de predição de popularidade devido a essa abordagem ter encontrado bons resultados nas áreas de *Visual Question Answer* (VQA)[2](ANTOL et al., 2015) e legenda em imagem (KARPATHY; LI, 2014). Wu et al. (2017) propôs um novo modelo de Redes Temporais de Contexto Profundo (DTCN), no qual, utiliza a informação sequencial de publicações anteriores em conjunto com o contexto temporal e múltiplas escalas de tempo. A partir dos dados de publicações anteriores foi criado um *embedding* de representação multimodal que é correlacionado com dois tipos de contextos temporais, Contexto Temporal Vizinho (NTC) e Contexto Temporal Periódico (PTC), para aprenderem as flutuações

[2] é uma área de pesquisa sobre a construção de um sistema de computador para responder às perguntas apresentadas em uma imagem e em uma linguagem natural.

da popularidade. O mecanismo de atenção é utilizado para considerar o impacto que cada um dos contextos temporais apresenta para popularidade. As análises realizadas sobre seu modelo DTCN apresentaram excelentes performances, porém, esse estudo foca apenas em entender a relevância da informação temporal para a predição de popularidade.

No estudo realizado por Zhang et al. (2018) foi apresentada um método em que o ambiente que caracteriza o usuário é descoberto. Esse ambiente representa o tipo de conteúdo que um determinado usuário normalmente tende a publicar. Para essa finalidade, foi utilizado um modelo de atenção dupla, que utiliza a técnicas de co-atenção introduzida por Wang, Sun e Chen (2019), para capturar melhores representações entre as informações textuais e visuais. A partir de análises estatísticas identificou qual o tipo de imagem ou título pode ajudar o usuário a alcançar um alto engajamento. Entretanto, esse tipo de abordagem não contempla a capacidade de generalização do modelo por focar em usuários específicos.

Na mesma linha de mecanismos de atenção, Zhang et al. (2018) apresentou um modelo multimodalidade que utiliza representações visuais e representações textuais guiadas por uma representação do usuário. Dessa forma, esse modelo é capaz de definir quais modalidades, textuais e visuais, são mais relevantes para um determinado usuário utilizando camadas *intra-attention* e *inter-attention*, baseados no modelo de atenção dupla, desenvolvido por Nam, Ha e Kim (2016), guiadas pelo usuário. A *intra-attention* é responsável por obter *embeddings* da representação textual e visual, já a *inter-attention*, é responsável por julgar a importância de cada *embedding* obtida na camada anterior. Apesar desse modelo superar modelos robustos no estado da arte de popularidade de predição, também não contempla a capacidade de generalização do modelo, impossibilitando a predição de conteúdos de usuários nunca visto antes pelo modelo.

Muitos dos estudos abordados negligenciam o problema de popularidade em aspectos de diversidade e capacidade de generalização. Modelos que utilizam contexto social negligenciam as características de diversidade dos usuários, em contrapartida, modelos que focam em usuários específicos acabam apresentando pouca capacidade de generalização. Nesse caso, os modelos apresentam uma solução parcial que atua apenas em alguns cenários específicos sem abranger a diversidade e capacidade generalização dos modelos. Além disso, nenhum dos estudos realizam análises detalhadas sobre as informações textuais contidas nas publicações, negligenciando a importância que essas informações podem agregar ao problema de popularidade.

Portanto, será proposta uma abordagem que se baseia no uso de multimodalidades que comtemple a diversidade e a capacidade de generalização, utilizando diferentes extratores de características textuais durante a etapa de treinamento.

3 REFERENCIAL TEÓRICO

Nas próximas seções são levantados os tópicos que auxiliaram no desenvolvimento desse estudo. Nas Seções 3.1 e 3.2 são abordados os métodos utilizados para extrair as modalidades visuais e textuais dos conteúdos, assim como, as representações textuais a nível de palavra e caractere. Na Seção 3.3 é introduzida a abordagem dos mecanismos de atenção que revolucionaram a tradução automática de idiomas. E por fim, na Seção 3.4 são abordas as técnicas de discretização de variáveis contínuas.

3.1 REDES NEURAIS CONVOLUCIONAIS

Redes Neurais Convolucionais (CNN), no mundo da aprendizagem de máquina, é considerado como um dos algoritmos mais populares Redes Neurais Profundas (DNN). As CNNs são consideradas uma das melhores técnicas para entender o conteúdo da imagem e apresentam resultados no estado da arte em reconhecimento de face, classificação de imagens, segmentação de imagens, detecção de objetos (ALOM et al., 2018) e (YANDONG HAO ZONGBO, 2016).

Entretanto, esse tipo de rede existe desde 1989 (LECUN et al., 1989), que propôs a primeira CNN multicamadas. Porém, se tornou popular apenas em 2012, com a arquitetura chamada de AlexNet, desenvolvido por Alex Krizhevsky (KRIZHEVSKY; SUTSKEVER; HINTON, 2012). Essa arquitetura foi submetida a um desafio de reconhecimento visual *ImageNet Large-Scale Visual Recognition Challenge* (ILSVRC) 2012, alcançando uma precisão de classificação de imagem substancialmente maior que os demais candidatos. Sua Rede Neural Convolucional foi treinada no banco de dados ImageNet com 1,2 milhão de imagens rotuladas utilizando aumento de dados, e estruturado com camadas modificadas, como $ReLU$ e Dropout.

Das muitas arquiteturas existentes para CNN, a AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), VGG (SIMONYAN; ZISSERMAN, 2014), a GoogLeNet com unidades de Inception (SZEGEDY et al., 2015), (SZEGEDY et al., 2017), a ResNet (HE et al., 2016), a Dense CNN (HUANG; LIU; WEINBERGER, 2016) e a FractalNet (LARSSON; MAIRE; SHAKHNAROVICH, 2016) são geralmente consideradas as arquiteturas mais populares. Cada arquitetura, além da complexidade, apresenta uma quantidade diferente de camadas e uma quantidade diferente de número de filtros utilizados na convolução. Algumas dessas arquiteturas são projetadas especialmente para análise de dados em grande escala, como GoogLeNet e ResNet, enquanto a rede VGG é considerada uma arquitetura geral.

Uma das grandes diferenças entre a CNN e os modelos tradicionais de aprendizagem de máquina é a sua capacidade de extrair recursos automaticamente durante o treinamento. As abordagens tradicionais de ML usam recursos obtidos manualmente, aplicando vários

algoritmos de extração e, em seguida, aplicando os algoritmos de aprendizado. No caso da CNN, e outros métodos de DNNs, os recursos são aprendidos automaticamente e são representados hierarquicamente em vários níveis. Além disso tudo, é possível usar um modelo treinado em um grande conjunto de dados e transferir seu conhecimento para um conjunto de dados menor. Essa técnica é denominada de *tranfer learning*, onde uma rede neural usada em uma tarefa específica (Ex.: uma rede com pesos treinados na *ImageNet* (SIMONYAN; ZISSERMAN, 2014)), é aproveitada e aplicada em outro domínio.

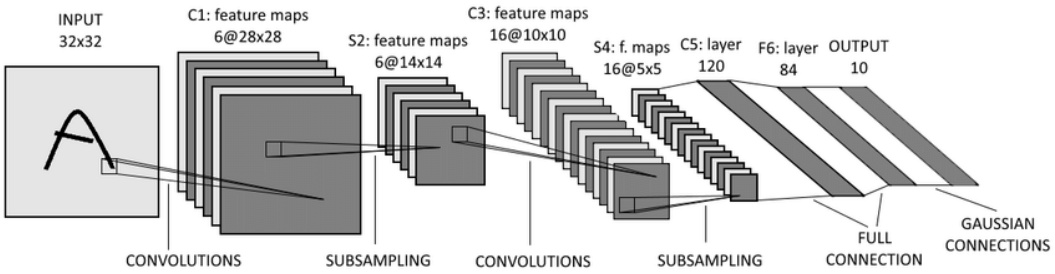


Figura 2 – Arquitetura do LeNet apresenta duas camadas convolucionais e duas camadas de subamostragem intercaladas para formar as primeiras quatro camadas. Camadas de ativação (não mostradas) são adicionadas após cada camada até F6. Duas camadas totalmente conectadas são anexadas após as últimas camadas de subamostragem para vetorizar as representações de imagem. A última e a camada de saída são compostas por unidades *Euclidean Radial Basis Function* (RBF) e realização a classificação para 10 classes.(**Fonte**: Krizhevsky, Sutskever e Hinton (2012))

Geralmente, uma arquitetura CNN típica compreende camadas alternadas de convolução e *pooling*, camadas que realizam a extração de característica, seguidas por uma ou mais camadas totalmente conectadas no final, como na Figura 2. A CNN é amplamente utilizada em etapas de extração e geração de recursos, onde, recursos de nível superior são derivados de recursos propagados de camadas de nível inferior. Enquanto esses recursos se propagam de uma camada para outra, as dimensões dos recursos são reduzidas a partir das operações de convolução e *pooling*. No entanto, o número de *feature maps* geralmente aumenta para representar melhor os recursos da imagem de entrada para garantir a precisão da classificação (KHAN et al., 2019).

3.1.1 Camada de Convolução

O principal papel da convolução é a extração de características da imagem de entrada, onde são aprendidas a conexão espacial entre os pixels. A camada convolucional é composta por um conjunto de núcleos convolucionais, denominados filtros. Esses filtros são matrizes com um conjunto específico de pesos, que são os elementos multiplicadores do filtro, e dividem a imagem em pequenos blocos, convolvendo-os (BOUVRIE, 2006). A operação de convolução é explicada pela seguinte equação:

$$\mathbf{F}_{\mathbf{l}}^{\mathbf{k}} = \left(\mathbf{I}_{\mathbf{x,y}} * \mathbf{K}_{\mathbf{l}}^{\mathbf{k}}\right)$$

Onde $\mathbf{I}_{\mathbf{x,y}}$ representa a imagem de entrada , sendo x, y a localidade espacial da imagem. $\mathbf{K}_{\mathbf{l}}^{\mathbf{k}}$ representa a l^{th} filtro convolucional da k^{th} camada. Dessa forma $\mathbf{F}_{\mathbf{l}}^{\mathbf{k}}$ é o *feature map* obtido pelo l^{th} filtro convolucional da k^{th} camada. A divisão da imagem em pequenos blocos ajuda a extrair valores de pixels correlacionados localmente. Um exemplo da aplicação desses filtros é demostrado na Figura 3, que contém um exemplo de um filtro sendo aplicado em uma imagem de apenas um canal. Nesse caso, a camada apresenta apenas um único filtro que opera sobre a matriz de pixels da imagem, preservando a relação espacial entre os pixels gerando assim um único canal de saída, que é definido como *feature map*.

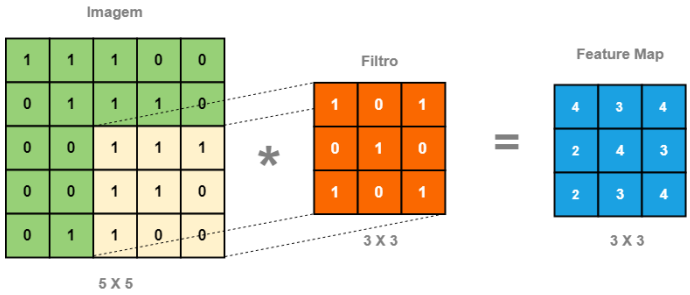


Figura 3 – Exemplo da aplicação de um filtro sobre uma imagem gerando assim o *feature map* da imagem.

Na Figura 4, contém um exemplo com múltiplos filtros operando sobre a imagem, ou seja, é evidente que diferentes filtros irão produzir diferentes *features maps* para uma mesma imagem dada como entrada. Nesse caso, se existirem mais filtros diferentes, mais características serão extraídas, e melhor a rede se torna em identificar padrões em imagens nunca vistas antes. Os filtros que são responsáveis por detectarem padrões nas imagens, a partir deles é possível identificar bordas, quinas, curvas, texturas e etc. A medida que são adicionadas mais camadas, os filtros se tornam mais sofisticados a ponto de extrair características de alto nível, não só detectando bordas e texturas como também olhos, face, orelhas e etc.

A quantidade de *feature maps* depende de três parâmetros que precisam ser definidos antes de inicializar o treinamento do modelo:

- ***Depth***: Corresponde a quantidade de filtros que são utilizados na operação de convolução. Como demostrado na Figura 4, a quantidade de *feature maps* gerado incrementa á medida que a quantidade de filtros são incrementados.

- ***Stride***: Se trata do número de pixels em que a o filtro é deslizado sobre a matriz. Quando o *stride* apresenta o valor de 1, o filtro é movido 1 pixel a cada passo da convolução, quando o *stride* tem o valor igual a 2, o filtro é movido 2 pixels a cada convolução. A Figura 5(a) apresenta um exemplo de uma convolução com *stride*

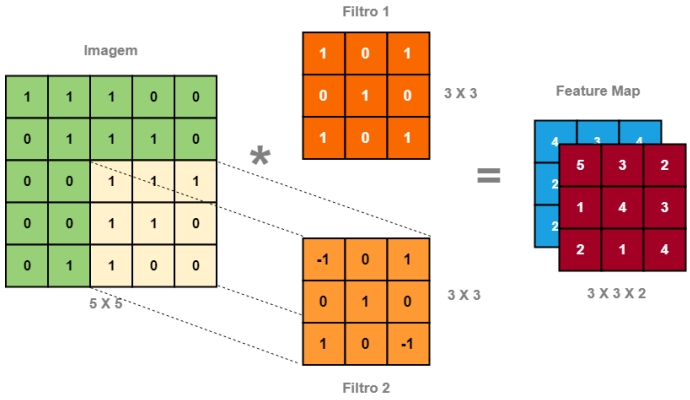
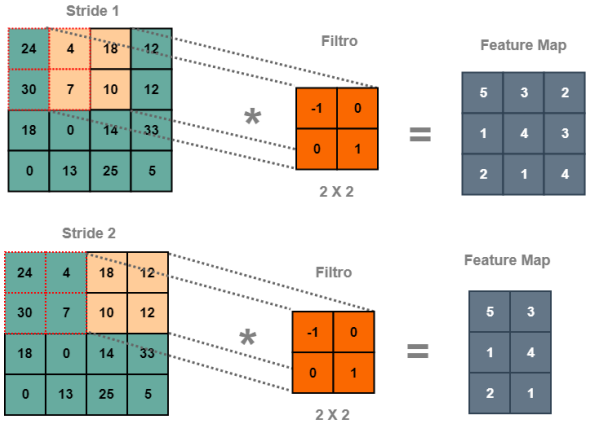


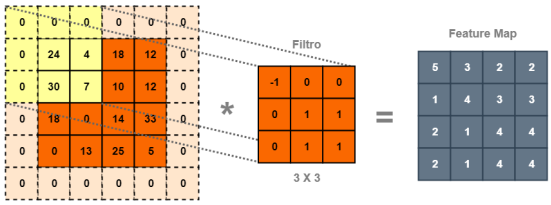
Figura 4 – Exemplo da aplicação de múltiplos filtros sobre uma mesma imagem, obtendo diferente *features maps*.

igual a 1 e outro com *stride* igual a 2, onde pode ser notado que á medida que se aumenta o *stride* a quantidade de características geradas no canal de saída vai diminuindo.

- ***Zero-padding***: Algumas vezes é necessário preencher a matriz de pixels com zeros ao seu redor, para que seja possível aplicar o filtro nos elementos vizinhos da nossa matriz de entrada. Isso permite controlar o tamanho do *feature map* gerado no canal de saída, Figura 5(b).



(a) Exemplos de um *stride* com valor de 1 pixel e outro com valor de 2 pixels.



(b) Exemplo da aplicação do *padding* sobre a imagem.

Figura 5 – Exemplo da aplicação do *stride* e do *padding*.

3.1.2 Camada de *Pooling*

Espacial *pooling*, também chamado de *subsampling* or *downsampling*, reduz a dimensionalidade de cada *feature map* obtida a partir da camada convolucional, retendo as informações mais importantes e reduzindo também a quantidade de parâmetros, consequentemente, reduz o consumo de memória nas camadas mais profundas. Geralmente os

métodos usados no *pooling* são o *avarage-polling* e o *max-polling*. Em (BOUREAU et al., 2010) foi feita uma comparação entre os dois, mostrando que o *max-polling* tem apresentado melhores resultados.

A forma mais comum da camada de *pooling* é uma matriz (2×2) aplicada com *stride* de 2. Dessa forma, quando o filtro do *pooling* percorre o *feature map*, é realizada a extração do maior elemento entre os vizinhos, se for utilizado o *max-pooling*. Caso seja utilizado o *avarage-pooling*, é realizada a média entre os vizinhos. A Figura 6 contém um exemplo de como é realizado o *avarage-pooling* e o *max- pooling* sobre uma imagem.

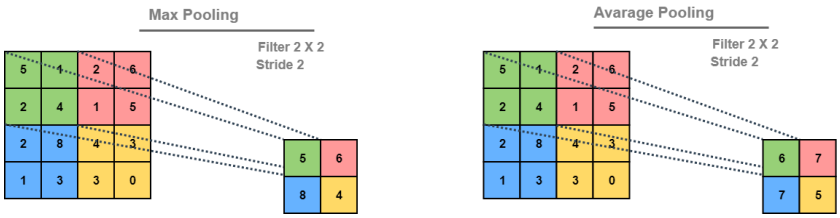


Figura 6 – Aplicação do *max-pooling* e do *avarage-pooling*.

A função da camada de *pooling* é reduzir progressivamente o tamanho espacial da representação da entrada. Em geral o *pooling* tem as seguintes características:

- Reduz a dimensionalidade da representação de entrada tornando-a mais manejável.
- Reduz o número de parâmetros e computações realizadas na rede, portanto, controla o overfiting.
- Torna a rede invariante a pequenas transformações, distorções e translações aplicada na imagem de entrada.

3.1.3 Camada de Ativação

A camada de ativação atua sobre as saídas de uma camada e ajuda a aprender um padrão complexo. O principal objetivo das funções de ativação é introduzir a não linearidade nas redes neurais. Sem funções de ativação, toda a rede neural seria uma transformação linear da entrada para a saída (FAN, 2016). A saída da camada de ativação é dada pela seguinte equação:

$$\mathbf{x^l} = f\left(\mathbf{x^{l-1}}\right)$$

$\mathbf{x^l}$ se trata da saída da camada de ativação l sobre o *feature map* $\mathbf{x^{l-1}}$ que introduziu a não linearidade. Na literatura existem diferentes funções de ativação, como *sigmoid*, *tanh* e $ReLU$. A $ReLU$ e suas variantes são preferidas em relação a outras ativações, pois ajudam a superar o problema do *vanishing* gradiante (HOCHREITER; SCHMIDHUBER, 1997).

3.1.3.1 *Rectified Linear Unit*

Rectified Linear Unit (ReLU), é uma operação não linear que é usada após a operação convolucional. A função do $ReLU$ é substituir todos os valores de pixels negativos, obtidos no *feature map*, para zero. Além disso, tem como proposta adicionar a não linearidade na CNN, já que a maioria dos dados do mundo real são de natureza não linear. A Figura 7 contém um exemplo da aplicação da $ReLU$ após obtenção do *feature map*, e também apresenta a função de ativação $ReLU$, que transforma todos os valores negativos em zero.

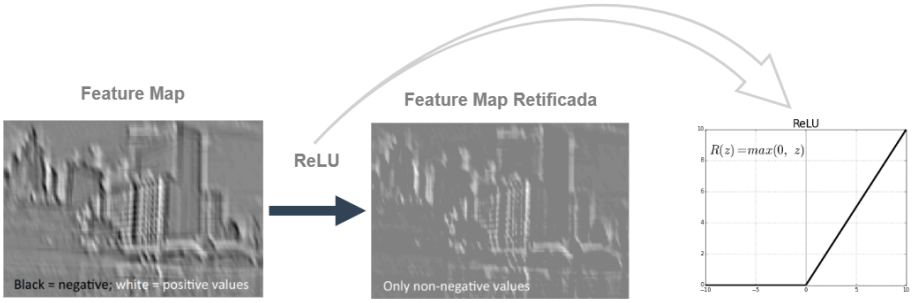


Figura 7 – Exemplo da aplicação do $ReLU$ sobre o *feature map* obtido de uma imagem. (**Fonte**: Haridas e Jyothi (2019))

3.1.4 Camadas Totalmente Conectadas

A camada totalmente conectada é usada principalmente no final da rede para fins de classificação. Essa camada recebe como entrada os recursos selecionados da camada anterior e faz uma combinação não linear, gerando novos recursos que serão usados para a classificação de dados. Em uma CNN típica, os mapas de características da última camada convolucional são vetorizados e dados como entrada para a camada totalmente conectada, que são seguidas por uma camada de classificação, que pode ser uma função de ativação *softmax*, ou outra camada totalmente conectada. A *softmax* é representada pela seguinte equação:

$$\mathbf{p_j} = \frac{e^{o_j}}{\sum_{k=1}^{N} e^{o_k}}$$

Assumindo que o vetor de saída seja $\mathbf{o} = [o_1, o_2, \ldots, o_N]$ e a saída do *softmax* seja $\mathbf{p} = [p_1, p_2, \ldots, p_N]$, sendo N a quantidade de rótulos que serão classificados pela rede neural, então temos que $\sum_{k=1}^{N} o_k = 1$. Portanto, a saída pode ser interpretada como distribuição de probabilidade em todas as classes sob um domínio de problema de classificação, pois, todas as unidades de saída do *softmax* somam 1.

3.2 PROCESSAMENTO DE LINGUAGEM NATURAL

Processamento de Linguagem Natural (NLP) concentra-se nas interações entre linguagem humana e computadores. A NLP é uma forma dos computadores analisarem e derivarem o significado da linguagem humana de uma maneira inteligente (LOPEZ; KALITA, 2017). A modelagem de linguagem é uma tarefa fundamental em inteligência artificial e NLP, com aplicações em reconhecimento de fala, geração de texto, tradução automática, análise de sentimento e reconhecimento de entidade (KIM, 2014).

Durante décadas, as abordagens de aprendizado de máquina visando problemas de NLP foram baseadas em recursos obtidos manualmente com dimensionalidade muito elevada. Esses recursos feitos manualmente são demorados e muitas vezes incompletos. Pois, a escolha de recursos é um processo completamente empírico, baseado principalmente em tentativa e erro, e a seleção de recursos é dependente de tarefas, o que implica pesquisa adicional para cada nova tarefa de NLP (XIE et al., 2018).

Nos últimos anos, NLP tem se beneficiado enormemente do ressurgimento das DNNs, devido ao seu alto desempenho com menor necessidade de recursos projetados manualmente. Os métodos de DNNs estão se tornando importantes em NLP devido ao seu sucesso em lidar com problemas complexos e permitir o aprendizado de representação automática de vários níveis (YOUNG et al., 2017).

Parte desse sucesso obtido pelas DNNs para tratar problema de NLP está relacionada ao *Word Embedding*. O *Word Embedding* é uma representação vetorial, com números reais, de palavras ou frases do vocabulário. Esses vetores conseguem capturar a similaridade entre as palavras, analogias e, devido à sua menor dimensionalidade, são rápidas e eficientes no processamento de tarefas centrais da NLP. Os *Word Embedding* foram responsáveis por resultados de última geração em uma ampla gama de tarefas de NLP (YOUNG et al., 2017).

Existem duas arquiteturas DNN que geralmente são aplicadas em problemas de NLP: Rede Neural Convolucional (CNN) (LECUN et al., 1989) e Rede Neural Recorrente (RNN) (ELMAN, 1990). CNNs e RNNs fornecem informações complementares para tarefas de classificação de texto. Escolher entre uma e outra arquitetura, para se obter o melhor desempenho, depende de quão importante é entender semanticamente de toda a sequência (YIN et al., 2017).

3.2.1 *Entity Embedding*

DNNs tem superado outros métodos de aprendizagem de máquina em várias áreas de pesquisa, como reconhecimento de imagem, classificação de áudio e processamento de linguagem natural entre outros de muitos exemplos na qual DNN apresenta resultados no estado da arte. Entretanto, todas essas áreas de pesquisa apresentam um fator em comum, todas elas lidam com dados não estruturados, como imagens, textos, vídeo e etc., que se

tratam de dados sem uma estrutura predefinida. De certa forma, DNN tem se tornado um padrão na escolha de um modelo que lide com dados não estruturados.

Mas alguns questionamentos eram levantados a partir da performance dos modelos de DNNs sobre dados estruturados (dados que são organizados em forma de tabelas que geralmente estão contidos em um banco de dados), pois, o *Gradiente Boosted Tree* era frequentemente mais utilizado por apresentar as melhores performances nas competições do Kaggle[1], como também na literatura acadêmica, do que modelos de redes neurais para operar sobre esses tipos de dados (CHEN; GUESTRIN, 2016). Além disso, redes neurais, devido a sua natureza contínua, limita sua aplicabilidade sobre variáveis categóricas, diferentemente, das árvores de decisão que não assumem nenhuma continuidade nas características (GUO; BERKHAHN, 2016).

Com o uso de *Entity Embedding*, as redes neurais começaram a apresentar resultados interessantes, mostrando que era possível alcançar e superar os desempenhos obtidos pelos modelos de *Boosted Tree* sobre dados estruturados. (BRÉBISSON et al., 2015) foi o ganhador da competição de predição de distância de taxi de passeio no Kaggle utilizando *Entity Embedding* sobre os metadados categóricos de cada taxi. Similarmente, (GUO; BERKHAHN, 2016) foram os terceiros colocados apresentando uma simples rede neural com *Entity Embedding* , de variáveis categóricas, apresentando uma estrutura menos complicada do que as utilizadas pelo primeiro e segundo lugar da competição.

Em resumo o *embedding* trata em representar uma categoria por um vetor de valores contínuos, por exemplo, a categoria $frio$ se torna algo como $[0.10, 0.15, 0.52, 0.47]$. Ou seja, cada categoria pertencente a uma variável discreta será mapeada para um vetor contínuo de valores que melhor a represente. Existem outros dois métodos comuns que geralmente são utilizados para processar dados categóricos em aprendizagem de máquina.

- ***One-hot encoding***: Cria um vetor binário com um tamanho da quantidade de categorias existentes, colocando inserindo 1 no índice referente a categoria, deixando os demais índices do vetor com 0, por exemplos, $frio \rightarrow [1, 0, 0]$, $normal \rightarrow [0, 1, 0]$ e $quente \rightarrow [0, 0, 1]$.

- ***Label enconding***: Associa um inteiro a cada categoria pertencente a variável categórica, tendo assim, $frio \rightarrow 1$, $normal \rightarrow 1$ e $quente \rightarrow 1$. Esse tipo de abordagem é muito adequado para modelos de árvores de decisão, mas não para modelos lineares

[1] Uma comunidade on-line de cientistas de dados e engenheiros de aprendizagem de máquina, de propriedade da Google LLC, na qual, permite que os usuários encontrem e publiquem conjuntos de dados, explorem e construam modelos, trabalhem com outros cientistas de dados e participem de competições para resolver desafios de ciência de dados.

3.2.1.1 Definição Matemática

A *Entity Embedding* mapeia cada categoria da variável discreta para um vetor:

$$e_i : x_i \mapsto \mathbf{x_i} \tag{3.1}$$

Esse tipo de mapeamento nada mais é do que uma camada linear extra de neurônios da rede neural sobre um *one-hot encoding*, tendo x_i o seguinte *one-hot encoding*:

$$u_i : x_i \mapsto \delta_{x_i\alpha} \tag{3.2}$$

Onde $\delta_{x_i\alpha}$ é o Kronecker delta:

$$\delta_{ij} = \begin{cases} 0, & se \;\; i \neq j \\ 1, & se \;\; i = j \end{cases} \tag{3.3}$$

os possíveis valores para α são os mesmos para x_i. Se m_i é o número de categorias da variável x_i , então $\delta_{x_i\alpha}$ é um vetor de tamanho m_i, onde o elemento não é zero quando $\alpha = x_i$.

A saída da camada linear extra de neurônios para um dada entrada x_i é definida como:

$$\mathbf{x_i} \equiv \sum_{\alpha}^{m_i} \mathbf{W}_\alpha \delta_{x_i\alpha} = w_{x_i} \tag{3.4}$$

onde $\mathbf{W}_\alpha \in \mathbb{R}^{d \times m_i}$ é o peso que conecta a camada do *one-hot encoding* a camada do *embedding*, onde $\mathbf{x_i} \in \mathbb{R}^d$. d é o valor da dimensionalidade do vetor que irá representar o valor discreto, que se trata de um hiperparâmetro. Dessa forma, é possível ver que o *embedding* nada mais é do que pesos aprendidos pela camada durante o treinamento do modelo. As vantagens de se obter o *embedding* são as seguintes:

- *Entity Embeddings* resolvem as desvantagens do *one-hot encondig*, pois, em um cenário em que a variável discreta apresenta muitas categorias serão gerados vetores que iram ocupar muito espaço na memória e que são computacionalmente ineficientes.
- Os *Embeddings* fornecem informação sobre a distância entre diferentes categorias. A beleza em usar *embeddings* é que o vetor que será relacionada a cada categoria é também treinado durante o treinamento da rede, utilizando a mesma função objetivo da rede. Este *embedding* treinado pode ser então visualizado fornecendo assim insights sobre cada categoria. (GUO; BERKHAHN, 2016) observou através da visualização do *embeddings* dos estados da Alemanha, agrupamentos semelhantes relacionados às localizações geográficas dos estados.
- O *embedding* treinado pode ser salvo e utilizado em um modelo de aprendizagem de máquina. Por exemplo esses *embedding* pode ser então usado no treinamento de modelo de *Random Forest* ou *Gradient Boosted Trees* para mapear os dados categóricos.

O crescimento do uso de redes neurais em problemas de processamento de linguagem natural é devido ao uso do *embbeding*, que para problemas de linguagem natural é definido como *Word Embedding*, que mapeiam palavras e frases em um vetor contínuo distribuído em um espaço semântico, permitindo assim que palavras similares fiquem próximas umas das outras (MIKOLOV et al., 2013), (BENGIO et al., 2003), (PENNINGTON; SOCHER; MANNING, 2014) como demostrado na Figura 8.

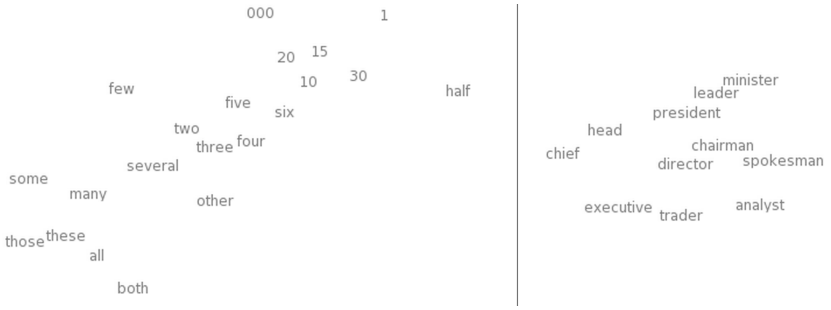


Figura 8 – Visualização dos *Word Embedding* pelo t-SNE. A esquerda: Região do Número; A direita: região de empregos. (**Fonte**: Turian, Ratinov e Bengio (2010)).

O que é mais interessante é, que não é apenas a distância entre as palavras que apresentam um significado, mas também, a direção da diferença entre os vetores, como foi observado por Bengio et al. (2003), onde as palavras aprendidas apresentam as seguintes relações.

$$King - Man \approx Queen - Woman$$

$$Paris - France \approx Rome - Italy$$

Existem várias abordagens para se aprender *Word Embedding*, a mais conhecida é a Word2Vec, que é baseado em um simples mais eficiente arquitetura que fornece propriedade semânticas interessantes (MIKOLOV et al., 2013), e o GloVe (PENNINGTON; SOCHER; MANNING, 2014), que combina matrizes globais de fatorização e uma janela de contexto local através de uma regressão bilinear. Porém, caso se trate de uma aplicação específica o interessante é aprender o *embedding* durante a fase de treinamento.

3.2.2 CNNs em processamento de linguagem natural

Quando ouvimos falar sobre as CNNs, normalmente pensamos em Visão Computacional. As CNNs foram responsáveis por grandes avanços na classificação de imagens. Mas recentemente, também começaram a ser aplicadas em problemas de Processamento de Linguagem Natural e alcançaram excelentes resultados (COLLOBERT et al., 2011).

A ideia por trás do uso de CNNs em NLP é fazer uso de sua capacidade de extrair recursos. No caso de NLP, se tratam de recursos de nível superior de palavras ou *n-grams*. Esses recursos são usados em várias tarefas de NLP, como análise de sentimento, resumo,

tradução automática e perguntas respondidas. Para extraí-los, são utilizados os filtros, que servem como detectores de *n-grams* (cada filtro procura por uma classe específica de *n-grams*), criando uma representação semântica da sentença de entrada.

As aplicações das CNNs em NLP foi iniciada por Collobert et al. (2011), Kalchbrenner, Grefenstette e Blunsom (2014), Santos e Gatti (2014) e Kim (2014). Essas abordagens alcançaram excelentes resultados, que mostraram que era possível codificar aspectos importantes da sintaxe e da semântica da linguagem. Tudo isso levou a uma enorme proliferação de redes baseadas na CNN para tratar tarefas tradicionais da NLP.

Em vez de pixels de imagem, a entrada para a maioria das tarefas de NLP são sentenças ou documentos representados como uma matriz. Cada linha da matriz corresponde a um *token*, normalmente uma palavra, mas pode ser um caractere. Ou seja, cada linha é vetor que representa uma palavra. Tipicamente, esses vetores são *embeddings* de palavras (representações de baixa dimensão), mas também podem ser vetores únicos que indexam a palavra em um vocabulário.

Na visão computacional, os filtros deslizam sobre manchas locais de uma imagem, mas em NLP os filtros deslizam sobre as linhas completas da matriz, onde cada linha representa uma palavra. Assim, a largura dos filtros é geralmente a mesma largura da matriz de entrada. A altura ou tamanho da região podem variar, na qual, janelas deslizantes com mais de 2 a 5 palavras por vez são o tamanho mais típico de serem aplicados sobre a matriz de palavras.

3.2.2.1 Modelagem da CNN para Sentenças

Para cada sentença, em que $\mathbf{w}_i \in \mathbb{R}^d$ representa o *Word Embedding* para a i^{th} palavra na sentença, onde d é a dimensão do *Word Embedding*. Dado que uma sentença tem n palavras, uma sentença pode ser representada pela matriz de *embedding* $\mathbf{W} \in \mathbb{R}^{n \times d}$. Visto que $\mathbf{w}_{i:i+j}$ se refere a concatenação dos vetores de sentenças com a matriz de *embedding* $\{\mathbf{w}_i, \mathbf{w}_{i+1}, \ldots, \mathbf{w_j}\}$. É realizada a operação de convolução na matriz de *embedding* de cada sentença. A convolução envolve um filtro $\mathbf{k} \in \mathbb{R}^{hd}$ que é aplicada em uma janela h de palavras para produzir novas características. Uma características nova $\mathbf{c}_i$ é gerada usando uma janela de palavras $\mathbf{w}_{i:i+h-1}$. Sendo $\mathbf{c}_i$ obtida por:

$$\mathbf{c}_i = f(\mathbf{w}_{i:i+h-1} * \mathbf{k} + \mathbf{b})$$

Aqui, $\mathbf{b} \in \mathbb{R}$ se trata do viés e f é uma função de ativação não linear. O filtro $\mathbf{k}$ é aplicado sobre todas as janelas possíveis usando os mesmos pesos, criando assim o seguinte *feature map*:

$$\mathbf{c} = [c_1, c_2, \ldots, c_{n-h+1}]$$

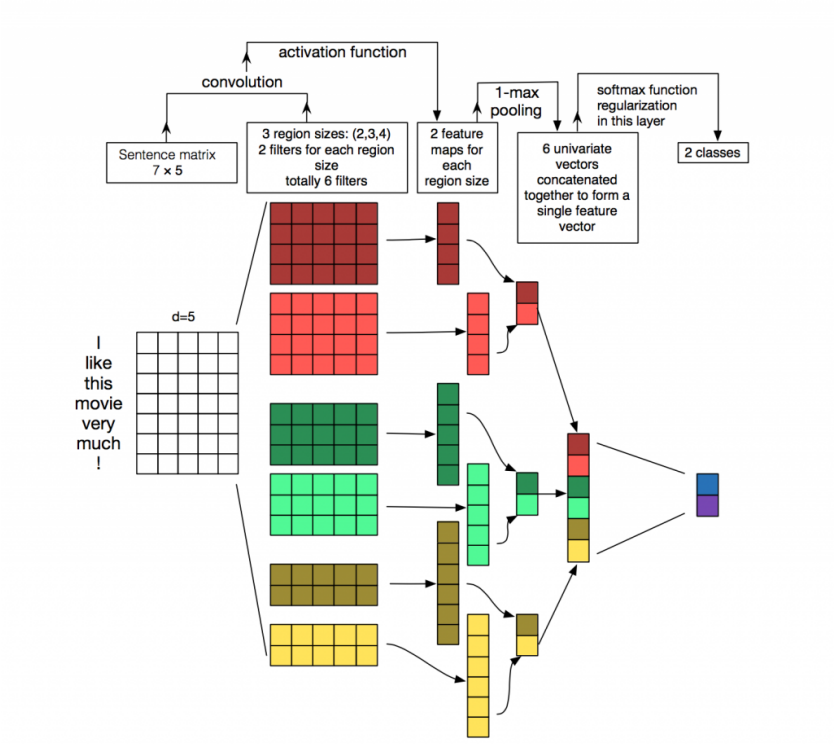


Figura 9 – Ilustração de uma arquitetura de Rede Neural Convolucional (CNN) para classificação de sentenças. Foram definidos três tamanhos de região de filtro: 2, 3 e 4, cada um dos quais tem 2 filtros. Cada filtro executa a convolução na matriz de sentenças e gera *features maps* de comprimentos variáveis. Em seguida, o agrupamento 1-max é realizado em cada *feature map*, ou seja, o maior número de cada *feature map* é registrado. Assim, um vetor de característica é gerado a partir de todos os seis *features maps*, e esses 6 recursos são concatenados para formar um vetor de característica para a penúltima camada. A camada *softmax* final recebe então esse vetor de recurso como entrada e o usa para classificar a sentença; (**Fonte**: Zhang e Wallace (2015)).

Então, vários filtros de diferentes tamanhos serão aplicados sobre a matriz de *embedding* obtida para cada sentença, onde cada filtro irá extrair um padrão específico de *n-gram*. A camada de convolução é seguida de uma camada de *max-pooling*, que opera sobre $\mathbf{c}$, tendo assim:

$$\hat{c} = max(\mathbf{c})$$

Dessa forma, a dimensionalidade da saída é reduzida, mantendo os recursos *n-gram* mais relevantes em toda a sentença. Cada filtro é agora capaz de extrair um recurso particular de qualquer lugar da sentença e adicioná-lo à representação final da sentença. O *embeddings* de cada palavra pode ser inicializada aleatoriamente e aprendido durante o treinamento ou pré-treinada sobre grandes corpos. A Figura 10 apresenta um exemplo da aplicação da CNN sobre um conjunto de palavras pertencentes a uma sentença.

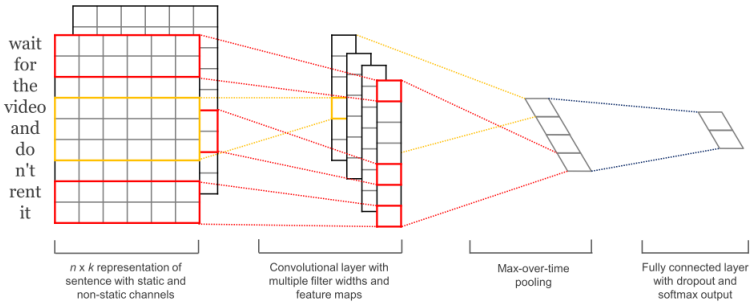


Figura 10 – Arquitetura de modelo com dois filtros para uma sentença . (**Fonte**: Kim (2014))

3.2.2.2 Modelagem da CNN a nível de caractere

Dado ζ como vocabulário de caracteres, d a dimensiona do *embedding* de caracteres e $\mathbf{Q} \in \mathbb{R}^{d \times |\zeta|}$ a matriz de caracteres *embedding*, sendo $|\zeta|$ o tamanho do vocabulário. Supondo que uma palavra $\mathbf{w}$ é composto de uma sequência de caracteres $[c_1, c_2, \ldots, c_l]$, onde l é o tamanho da palavra $\mathbf{w}$. Tem-se então, que a representação a nível de caractere da palavra $\mathbf{w}$ é dada pela matriz $\mathbf{C^w} \in \mathbb{R}^{d \times l}$, onde a j^{th} coluna corresponde ao caractere *embedding* $\mathbf{c}_j$. Então, a convolução é aplicada entre $\mathbf{C^w}$ e o filtro $\mathbf{H} \in \mathbb{R}^{d \times k}$ de tamanho k. Sendo assim:

$$\mathbf{f^w} = f(\mathbf{C^w}_{[i:i+k-1]} * \mathbf{H} + \mathbf{b})$$

$$\mathbf{y}^w = max\,(\mathbf{f^w})$$

Onde, $\mathbf{C^w}_{[i:i+k-1]}$ se trata da i^{th} até $i + k - 1^{th}$ coluna de $\mathbf{C^w}$ e $\mathbf{f^w}$ contém todas as caracteristicas obtidas pelo filtro H sobre $\mathbf{C^w}$. Então, $\mathbf{y}^w$ é característica mais importante de um dado filtro sobre $\mathbf{C^w}$. O filtro está essencialmente capturando caracteres de *n-gram*, onde o tamanho do *n-gram* corresponde ao tamanho do filtro. Esse exemplo apresenta o processo sobre uma única palavra utilizando um único filtro, geral são utilizados múltiplos filtros. A Figura 11 apresenta a aplicação da CNN a nível de caractere.

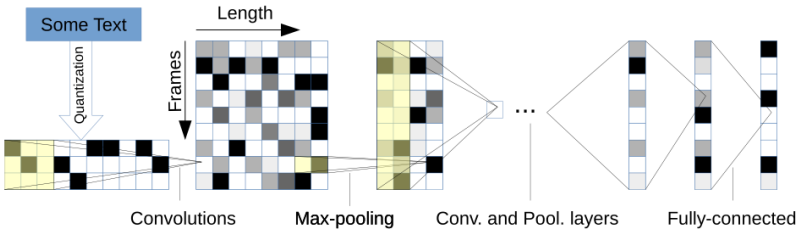


Figura 11 – Exemplo da arquitetura utilizada para extrair características a nível de caracteres. (**Fonte**: Zhang, Zhao e LeCun (2015))

3.2.3 Redes Neurais Recorrentes

A compreensão de uma frase ou uma palavra, quando se está lendo um livro, artigo, revista etc., é dada a partir do entendimento e conhecimento adquirido a partir de palavras ou frases anteriores. A uma persistência entre a informação anterior com a atual, não é simplesmente jogado tudo fora e começado do zero. Métodos tradicionais de redes neurais, como DNN e CNN, não apresentam essa capacidade de capturar a natureza sequencial inerente presenta na linguagem e operar sobre informações sequências ao longo do tempo.

Rede Neural Recorrente (RNN), que tiveram seu conceito introduzido em 1982 segundo Sathasivam e Abdullah (2008), são as únicas que apresentam essa capacidade, pois, apresentam uma modelagem sequencial podendo operar sobre dados sequências ao longo do tempo. As RNNs podem ser pensadas como múltiplas cópias de uma mesma rede onde cada cópia passa uma informação para sua sucessora, como demostrado na Figura 12.

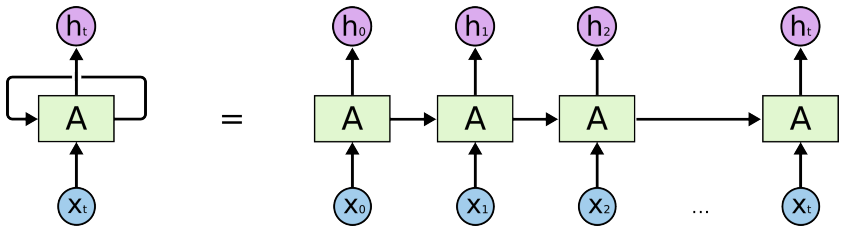


Figura 12 – Uma Rede Neural Recorrente "desenrolada". (**Fonte**: Olah (2015)) .

A arquitetura desenvolvida por Elman (1990), que se trata de uma simples RNN, apresenta uma rede de três camadas, que é desdobrada ao longo do tempo para acomodar a sequência inteira, onde $\mathbf{x}_t$ é tomado com a entrada para a rede na etapa t e $\mathbf{h}_t$ representa o estado oculto na mesma etapa de tempo. Sendo matematicamente expressada como:

$$\mathbf{h}_t = \sigma_h \left(\mathbf{W_h} \mathbf{x}_t + \mathbf{U_h} \mathbf{h}_{t-1} + b_h \right) \tag{3.5}$$

$$\mathbf{y}_t = \sigma_y \left(\mathbf{W_y} \mathbf{h}_t + b_y \right) \tag{3.6}$$

onde, $\mathbf{h}_t$ é calculado utilizando a entrada atual no tempo t e o estado oculto anterior no tempo $t-1$. E $\mathbf{y}_t$ o vetor de saída da RNN. A função σ é uma transformação não linear, tal como *tanh*, *ReLU*. $\mathbf{W}$ e $\mathbf{U}$ são matrizes de pesos que são compartilhados ao longo do tempo e b o vetor de viés.

Devido a sua natureza sequencial, há um grande sucesso ao ser aplicada a uma variedade de problemas de processamento de linguagem natural, como reconhecimento de voz, tradução, legenda em imagens, pergunta e resposta e etc. Apesar da RNN apresentar diferentes variações e versões (ELMAN, 1990) e (JORDAN, 1986), a essência desse seu sucesso é o uso da *Long Short Term Memory* (LSTM) introduzido por Gers e Schmidhuber (2000),

que tratam o problema em lidar com dependências de longo prazo. Esse problema foi explorado por Bengio, Simard e Frasconi (1994) e se trata do *vanishing gradient*, quando a rede "esquece" o que aconteceu anteriormente e não consegue propagar as dependências através da sequência inteira.

Mesmo as RNNs apresentando o recurso de conectar informações prévias com informações que vem posteriormente, por exemplo, usar os frames mais recentes de um vídeo pode ajudar a compreender qual será o próximo frame. Porém, nem sempre se é utilizada a informação mais recente para ajudar a prever o que vem a seguir. Em alguns casos é necessário deter de informações prévias mais antigas para entender melhor o contexto, e à medida que essa lacuna cresce, RNNs se tornam-se incapazes de aprender a conectar as informações, apresentando problemas em lidar com dependências de longo prazo.

3.2.3.1 *Long Short-Term Memory*

Os LSTMs são explicitamente projetados para evitar o problema de dependência de longo prazo. Guardar informações por longos períodos de tempo é praticamente o seu comportamento padrão. Em RNNs padrão, o módulo apresentado em cada etapa apresenta uma estrutura muito simplificada, apresentando uma única camada de rede neural, como apresentado na Figura 13. Entretanto, LSTMs possuem um módulo com uma estrutura diferente, como na Figura 14, apresentando quatro camadas de redes neurais, onde cada uma apresenta uma particularidade dentro da estrutura.

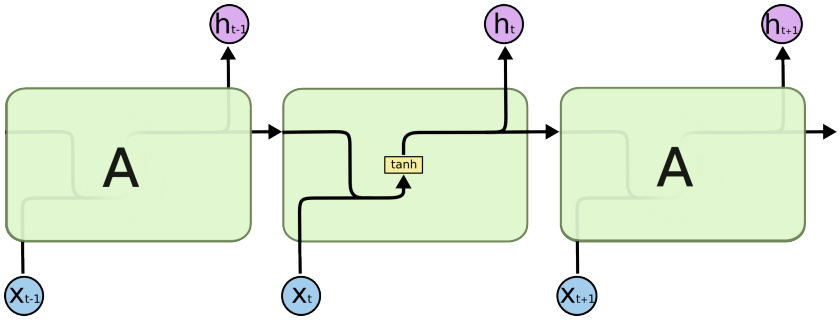


Figura 13 – O módulo de repetição em um RNN padrão contém uma única camada. (**Fonte**: Olah (2015)) .

A ideia chave das redes LSTMs é o estado da célula, a linha horizontal que passa pela parte superior no topo da Figura 14. É possível remover ou adicionar informações ao estado da célula, a partir de estruturas chamadas portas, *input-gate* ($\mathbf{i}_t$) , *forget-gate* ($\mathbf{f}_t$) e *output-gate* ($\mathbf{o}_t$), que servem para regular o que será adicionado ou removido.

O *forget-gate* decide quais informações serão descartadas no estado da célula a partir de uma função sigmoide.

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_\mathbf{f} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_\mathbf{f}\right) \tag{3.7}$$

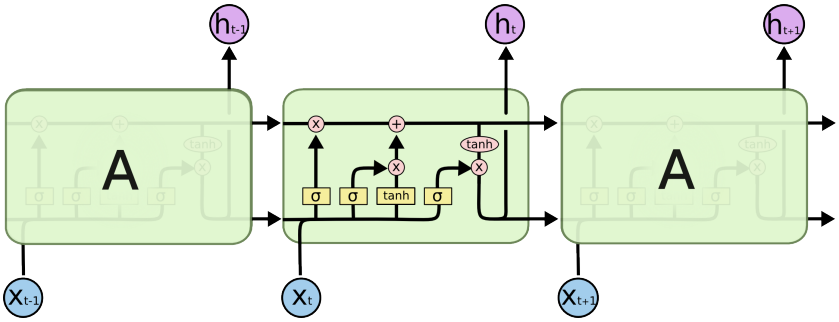
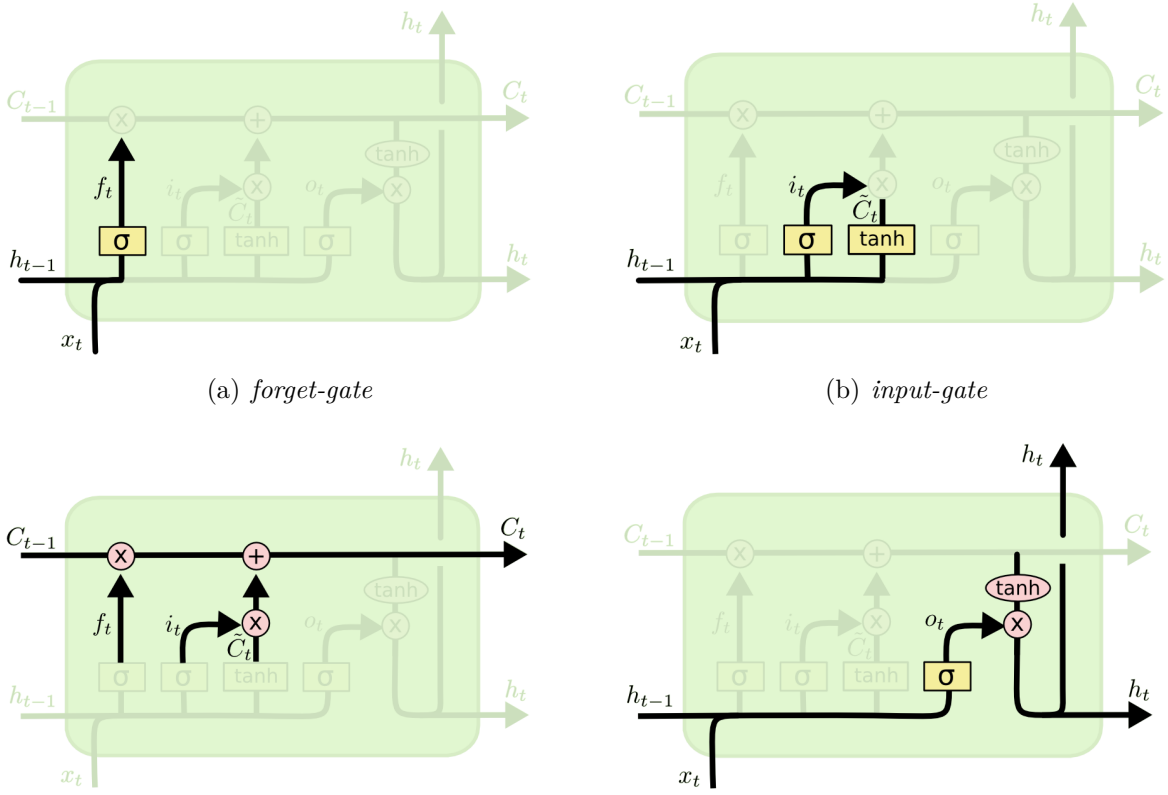


Figura 14 – Estrutura do módulo do LSTM que contém módulo de repetição com quatro camadas de interação . (**Fonte**: Olah (2015))



(a) *forget-gate* (b) *input-gate*

(c) update-gate (d) *output-gate*

Figura 15 – As quatro camadas do módulo de repetição da LSTM. (**Fonte**: Olah (2015))

O *forget-gate*, Figura 15(a), analisa o $\mathbf{h}_{t-1}$ e $\mathbf{x}_t$ e gera valores entre 0 e 1, definindo assim o quanto de informação em $\mathbf{C}_{t-1}$ irá permanecer. O *input-gate*, Figura 15(b), define o quanto de informação nova $\mathbf{x}_t$ será adicionada no estado da célula, utilizando também uma sigmoide.

$$\mathbf{i}_t = \sigma\left(\mathbf{W}_i \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b_i}\right) \tag{3.8}$$

$$\tilde{\mathbf{C}}_t = \tanh\left(\mathbf{W_C} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b_C}\right) \tag{3.9}$$

A camada $\hat{\mathbf{C}}_t$ apresenta os valores candidatos a serem adicionados no estado da célula que sofrem uma transformação não linear. A partir dessas três camadas a célula de estado $\mathbf{C}_{t-1}$ é atualizada para $\mathbf{C}_t$, Figura 15(c), através da seguinte formula.

$$\mathbf{C}_t = \mathbf{f}_t * \mathbf{C}_{t-1} + \mathbf{i}_t * \tilde{\mathbf{C}}_t \tag{3.10}$$

A última camada, a *output-gate*, decide o quanto de informação do estado da célula vai ser entregue ao estado oculto $\mathbf{h_c}$ através da sigmoide, Figura 15(d):

$$\mathbf{o}_t = \sigma\left(\mathbf{W_o} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b_o}\right) \tag{3.11}$$

$$\mathbf{h}_t = \mathbf{o}_t * tanh\left(\mathbf{C}_t\right) \tag{3.12}$$

Os LSTMs, ao tratar problemas que envolve dados textuais, como tradução automática de idiomas, classificação de texto e análise de sentimento, demostraram resultados interessantes. Wang et al. (2015) propôs a codificação de Tweets inteiros com o LSTM, utilizando o *Word Embedding* como codificador de cada palavra, cujo o último estado oculto é usado para prever a polaridade do sentimento. Essa estratégia simples mostrou-se competitiva em relação a outras estruturas mais complexas de DNN.

Em (SUTSKEVER; VINYALS; LE, 2014), os autores propuseram uma estrutura geral de codificador-decodificador, definida como *seq2seq*, que mapeia uma sequência para outra sequência. No caso da tradução automática de idiomas, a LSTM é usada para codificar a sequência de entrada, e o vetor resultante, também chamado de vetor de contexto, obtido pelo último estado oculto, é usado como fonte de informação para o estado inicial do decodificador, que também é uma LSTM. Durante a inferência, o decodificador gera *tokens* de um por um, enquanto atualiza seu estado oculto com o último *token* gerado.

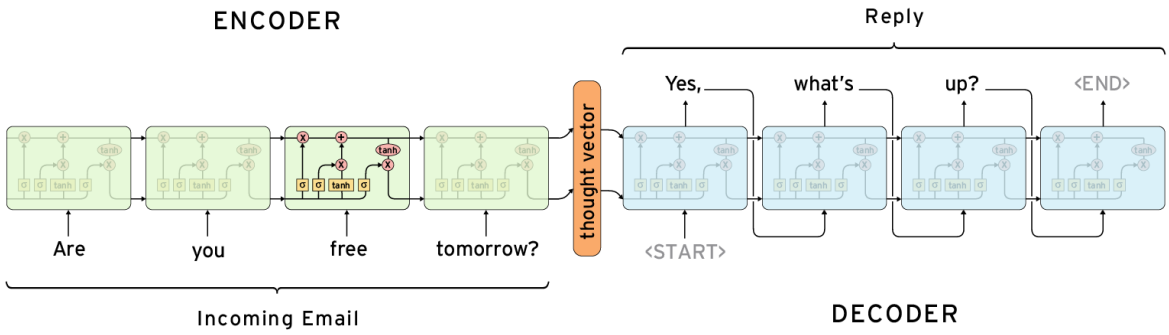


Figura 16 – Arqutietura *Seq2Sec* (codificador-decodificador). (**Fonte**: Sutskever, Vinyals e Le (2014))

3.3 MECANISMOS DE ATENÇÃO

Atenção, no contexto de redes profundas, significa se concentrar seletivamente em uma pequena parte dos dados gerais. Os mecanismos de atenção foram usados principalmente no campo da imagem visual por Posner e Petersen (1990). Mas só se tornaram populares a partir da publicação do artigo Modelos recorrentes de atenção visual desenvolvido pela equipe da Google (MNIH et al., 2014). Esse artigo demostra a aplicação dos mecanismos de atenção para a classificação de imagem utilizando RNN.

Entretanto, o sucesso dos mecanismo de atenção se deram a partir do seu uso no campo do Processamento de Linguagem Natural. Mnih et al. (2014) experimentou pela primeira vez os Mecanismos de Atenção no campo da NLP para tarefas de tradução automática. A partir desse dia em diante os Mecanismos de Atenção tornaram-se comuns nas tarefas de NLP baseadas em redes neurais como RNNs e CNNs.

Os mecanismos de atenção imitam o mecanismo da visão humana, dando um maior foco em diferentes regiões de uma imagem ou em algumas palavras presentes em um texto. Quando o mecanismo de visão humana detecta um item, ele foca apenas em determinadas regiões de acordo com suas necessidades. Quando uma pessoa percebe que o objeto ao qual deseja prestar atenção aparece geralmente em uma parte específica de uma cena, ela aprende que, no futuro, o objeto ficará nessa parte e tenderá a concentrar sua atenção nessa área.

Figura 17 – Uma palavra apresenta níveis de atenção diferentes para diferentes palavras.

Da mesma forma, pode-se explicar a relação entre palavras em uma frase ou contexto próximo. Na Figura 19, quando se encontra a palavra “dirigindo” no texto, espera-se encontrar uma palavra de algum veículo muito em breve. O termo de cor ("preto") remete a uma característica do carro, mas não tanto quanto a palavra “dirigindo”. Devido a essa particularidade, os mecanismos de atenção tiveram um grande sucesso para tratar problemas de tradução automática. O modelo sequência-a-sequência (*seq2seq*), que opera sobre problemas de tradução automática entre vários idiomas, de realizar a transformação de áudio para texto, de geração de diálogos a partir de perguntas e repostas e etc., foi introduzida por Sutskever, Vinyals e Le (2014). De um modo geral, visa transformar uma sequência de entrada para uma nova sequência de saída. O modelo *seq2seq* normalmente

possui uma arquitetura de codificador-decodificador, composta de:

- Um codificador processa a sequência de entrada, geralmente a partir de uma RNN, e compacta a informação em um vetor de contexto de um comprimento fixo. Essa representação apresenta, em tese, um resumo do significado de toda a sequência da como entrada para o modelo.
- Um decodificador é inicializado com o último estado da rede recorrente do codificador para transformar em uma saída que atenda o problema em questão. Esse último estado representa o contexto obtido pelo codificador.

Um problema potencial que essa estrutura do codificador-decodificador enfrenta é que o codificador às vezes é forçado a codificar informações que podem não ser totalmente relevantes para a tarefa em questão. Além disso, há também um problema em tratar sequências longas, pois, a primeira parte da sequência pode ser esquecida depois que todo o processamento sobre a entrada seja concluído.

O mecanismo de atenção ajuda a memorizar longas sentenças e escolher seletivamente alguns itens de entrada que são mais relevantes e estejam mais correlacionados a tarefa em questão. Nesse caso, em vez de criar um único vetor de contexto a partir do último estado oculto do codificador, para ser dado como entrada para o decodificador, é gerado um vetor de contexto para cada saída, onde cada vetor de contexto irá apresentar a relação de quais entradas, presentes no codificador, estão mais correlacionadas com uma saída específica do decodificador.

3.3.1 Definição Matemática

Dada uma sequência $\mathbf{x}$ de tamanho n que tem como saída uma sequência $\mathbf{y}$ de tamanho m:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n]$$

$$\mathbf{y} = [y_1, y_2, \ldots, y_m]$$

Dado que o codificador é uma RNN Bidirecional, aplicada sobre a sequência $\mathbf{x}$, obtemos os estados escondidos de avanço $\overrightarrow{\mathbf{h}_i}$ e os estados escondidos de retorno $\overleftarrow{\mathbf{h}_i}$ da rede bidirecional. Essas saídas são concatenadas, obtendo o seguinte vetor:

$$\mathbf{h} = \left[\overrightarrow{\mathbf{h}_i} : \overleftarrow{\mathbf{h}_i}\right], i = 1, \ldots, n$$

O decodificador tem o estado escondido $\mathbf{s}_t = f\left(\mathbf{s}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}_t\right)$ para cada saída t, onde $t = 1, \ldots, m$. $\mathbf{s}_{t-1}$ se trata da camada escondida $t-1$, do decodificador, e $\mathbf{c}_t$ é o vetor de contexto. $\mathbf{c}_t$ é a soma dos estados escondidos da sequência de entrada $\mathbf{x}$ ponderada pelos escores de alinhamento.

$$\mathbf{c}_t = \sum_{i=1}^{n} \alpha_{t,i} \mathbf{h}_i$$

$$\alpha_{t,i} = align(y_t, x_i) = \frac{\exp(score(\mathbf{s}_{t-1}, \mathbf{h}_i)))}{\sum_{j=1}^{n} \exp(score(\mathbf{s}_{t-1}, \mathbf{h}_j)))}$$

Onde $\alpha_{t,i}$ define quão bem uma saída $\mathbf{y}_t$ está correlacionada com a entrada $\mathbf{x}_i$, e $align(\mathbf{y}_t, \mathbf{x}_i)$ atribui uma pontuação $\alpha_{t,i}$ ao par de entrada na posição i e saída na posição t, $(\mathbf{y}_t, \mathbf{x}_i)$. O conjunto de $\{\alpha_{t,i}\}$ são pesos que definem o quanto que cada estado oculto, proveniente do codificador, deve ser considerado para cada saída t. No artigo de Bahdanau, Cho e Bengio (2014), a pontuação de alinhamento α é parametrizada por uma rede neural treinada em conjunto com outras partes do modelo. A função de pontuação $score(\mathbf{s}_{t-1}, \mathbf{h}_j)$ é, portanto, da seguinte forma:

$$score(\mathbf{s}_{t-1}, \mathbf{h}_j) = \mathbf{v_a} \tanh(\mathbf{W_a} [\mathbf{s}_t : \mathbf{h}_i])$$

onde tanh é a função de ativação não linear, $\mathbf{v_a}$ e $\mathbf{W_a}$ são as matrizes de pesos que serão aprendidas.

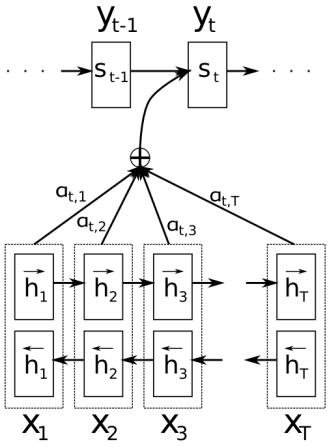


Figura 18 – O modelo codificador-decodificador com mecanismo de atenção aditiva. (**Fonte**: Bahdanau, Cho e Bengio (2014))

Devido a grande melhoria da atenção na tradução automática, o mecanismo de atenção logo se estendeu ao campo da visão computacional (XU et al., 2015) e os pesquisadores começaram a explorar várias outras formas e aplicações (LUONG; PHAM; MANNING, 2015) e (BRITZ et al., 2017). Dessa forma, Nam, Ha e Kim (2016) propôs as *Redes de Atenção Dupla* (DANs) que alavancam conjuntamente mecanismos de atenção visual e textual para capturar interação refinada entre visão e linguagem. DANs coletam informações essenciais de ambas as modalidades e permite que as atenções visuais e textuais se auxiliam na extração das melhores representações durante a inferência colaborativa, o que é útil para tarefas como VQA. Esse método realiza atenções visuais e textuais simultaneamente através de múltiplas etapas e reúne as informações necessárias de ambas as modalidades.

Então, para inferir a Atenção visual, no intuito de atender certas partes da imagem de entrada, é dada uma imagem I representada por $\{v_1, \ldots, v_N\}$, onde N é número de regiões da imagem. Supondo que a cada etapa k um contexto visual $\mathbf{v}^{(k)}$ seja dado por:

$$\mathbf{v}^{(k)} = \mathbf{V_{Att}}\left(\{\mathbf{v}_n\}_{n=1}^{N}, \mathbf{m}_{\mathbf{v}}^{(k-1)}\right)$$

onde $\mathbf{m}_{\mathbf{v}}^{(k-1)}$ é vetor de memória conjunta que acumula as informações visuais e textuais que foram atendidas até o passo $k-1$. Os pesos de atenção $\left\{\alpha_{v,n}^{(k)}\right\}_{n=1}^{N}$ para cada região da imagem são computados a partir de duas camadas de rede neural *feedfoward* e uma função *softmax*:

$$\mathbf{h}_{\mathbf{v},n}^{(k)} = \tanh\left(\mathbf{W}_{\mathbf{v}}^{(k)}\mathbf{v}_n\right) \odot \tanh\left(\mathbf{W}_{\mathbf{v},\mathbf{m}}^{(k)}\mathbf{m}_{\mathbf{v}}^{(k-1)}\right)$$

$$\alpha_{v,n}^{(k)} = softmax\left(\mathbf{W}_{\mathbf{v},\mathbf{h}}^{(k)}\mathbf{h}_{\mathbf{v},n}^{(k)}\right)$$

$$\mathbf{v}^{(k)} = \sum_{n=1}^{N} \alpha_{v,n}^{(k)}\mathbf{v}_n$$

onde $\mathbf{W}_{\mathbf{v}}^{(k)}$, $\mathbf{W}_{\mathbf{v},\mathbf{m}}^{(k)}$ e $\mathbf{W}_{\mathbf{v},\mathbf{h}}^{(k)}$ são os parâmetros da rede. $\mathbf{h}_{\mathbf{v},n}^{(k)}$ é o estado escondido e $\odot$ é o elemento de multiplicação. Para a atenção textual, temos que para um conjunto de vetores de recursos $\{\mathbf{u_1}, \ldots, \mathbf{u_T}\}$, onde $\mathbf{u_T}$ codifica a semântica do t^{th} palavra no contexto de toda a frase. Então, o vetor de contexto textual $\mathbf{u}^{(k)}$ é obtido da seguinte maneira:

$$\mathbf{u}^{(k)} = \mathbf{T_{Att}}\left(\{\mathbf{u}_t\}_{t=1}^{T}, \mathbf{m}_{\mathbf{u}}^{(k-1)}\right)$$

onde $\mathbf{m}_{\mathbf{u}}^{(k-1)}$ é um vetor de memória que codifica as informações que foram atendidas até a etapa $k-1$. Os pessoas de atenção $\left\{\alpha_{u,t}^{(k)}\right\}_{t=1}^{T}$ para cada palavra inserida na sentença são computadas a partir de duas camadas de rede neural *feedfoward* e uma função *softmax*:

$$\mathbf{h}_{\mathbf{u},t}^{(k)} = \tanh\left(\mathbf{W}_{\mathbf{u}}^{(k)}\mathbf{u}_t\right) \odot \tanh\left(\mathbf{W}_{\mathbf{u},\mathbf{m}}^{(k)}\mathbf{m}_{\mathbf{u}}^{(k-1)}\right)$$

$$\alpha_{u,t}^{(k)} = softmax\left(\mathbf{W}_{\mathbf{u},\mathbf{h}}^{(k)}\mathbf{h}_{\mathbf{u},t}^{(k)}\right)$$

$$\mathbf{u}^{(k)} = \sum_{t=1}^{T} \alpha_{u,t}^{(k)}\mathbf{u}_t$$

onde $\mathbf{W}_{\mathbf{u}}^{(k)}$, $\mathbf{W}_{\mathbf{u},\mathbf{m}}^{(k)}$ e $\mathbf{W}_{\mathbf{u},\mathbf{h}}^{(k)}$ são os parâmetros da rede. $\mathbf{h}_{\mathbf{u},t}^{(k)}$ é o estado escondido e $\odot$ é o elemento de multiplicação.

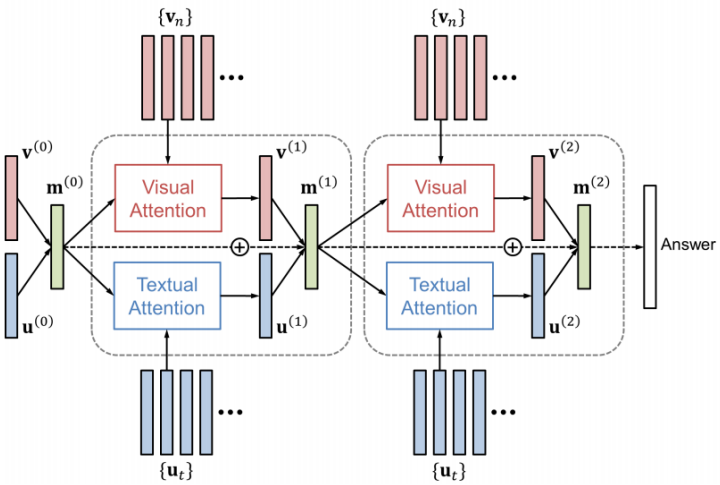


Figura 19 – Arqutietura DAN com $k = 2$. (**Fonte**: Nam, Ha e Kim (2016))

3.4 DISCRETIZAÇÃO DE VARIÁVEIS CONTÍNUAS

A discretização é uma das técnicas básicas de redução de dados que tem recebido uma maior atenção em pesquisas nos últimos anos (GARCIA et al., 2013) e tem se tornado uma das técnicas mais utilizadas em mineração de dados, uma vez que transforma dados quantitativos em dados qualitativos, que nada mais é do que transformar dados numéricos em discretos, possibilitando, a partir dos dados qualitativos, se ter uma maior interpretação dos dados e extrair um maior conhecimento sobre os mesmos.

A discretização é um processo em que há uma divisão de uma variável contínua em intervalos, onde cada intervalo é rotulado como um valor discreto, tendo assim um mapeamento do conjunto de valores numéricos de uma variável contínua para um conjunto de valores discretos, dessa forma há uma redução do conjunto numérico para um subconjunto de valores discretos. Uma variável contínua A é discretizada pelo o algoritmo em um conjunto m de valores discretos, então, assumindo um conjunto de dados de N exemplos:

$$D = \{[d_0, d_1], (d_1, d_2], \ldots, (d_{m-1}, d_m]\} \tag{3.13}$$

Onde d_0 é o valor mínimo e d_m é o valor máximo presente na variável contínua e $d_i < d_i + 1$ para $i = 0, 1, \ldots, m - 1$. Dessa forma, D é definido como esquema de discretização de uma variável contínua A e $P = d_1, d_2, \ldots, d_{m-1}$ é o conjunto de *cut-points* (pontos de corte). O termo "*cut-point*", também conhecido como split-point, se refere a um valor real que divide um conjunto de valores reais em dois intervalos, um intervalo é menor ou igual ao *cut-point* e o outro intervalo é maior que o *cut-point*, por exemplo, um intervalo contínuo $[a, b]$ é particionado em $[a, c]$ e $(c, b]$, logo, o valor c, seria o *cut-point*.

Segundo Hemada e Lakshmi (2013) existem várias vantagens em se utilizar valores discretos ao invés de valores contínuos:

- A discretização reduz o número de variáveis contínuas trazendo assim uma pequena demanda para o armazenamento do sistema;
- Variáveis discretas são mais fáceis de serem interpretadas, usadas e explicadas agregando mais conhecimento do que variáveis contínuas;
- A discretização faz com que a aprendizagem seja mais rápida e precisa;
- Alguns algoritmos de aprendizado só podem lidar com dados discretos.

Além disso, Chlebus e Nguyen (1998) e Richeldi e Rossotto (1995) demostraram que alguns algoritmos que são capazes de lidar com variáveis contínuas apresentaram uma aprendizagem menos eficiente e eficaz ao tratar dados contínuos do que dados discretizados. Os algoritmos de discretização podem ser categorizados em supervisionados e não supervisionados. Os supervisionados utilizam informações de classe para guiar o processo de discretização, utilizando heurísticas para determinar os melhores *cut-points*, como o método Ent-MDLP (FAYYAD; IRANI, 1993), que utiliza a entropia para encontrar *cut-point* em potencial.

Os não supervisionados, não necessitam de instâncias rotuladas, uma vez que obter instâncias rotuladas conduz a outras dificuldades encontradas em problemas de aprendizagem de máquina, e como esse trabalho trata de um problema de regressão, não teria como ser aplicados os métodos supervisionados, uma vez que um discretizador supervisionado só pode ser aplicado sobre problema supervisionados (GARCIA et al., 2013). *Equal-Width* e *Equal-Frequency* (Wong; Chiu, 1987), são dois representantes de algoritmos de discretização não supervisionado.

A aplicação da discretização pode ser feita de maneira dinâmica ou estática. O método dinâmico discretiza os valores duração o treinamento do classificador, já o estático, discretiza os valores antes de iniciar o treinamento do classificador. Entretanto, quase todos os discretizadores conhecidos são estáticos (BERZAL et al., 2004). Os métodos de discretização também podem ser globais ou locais, isto é, para tomar uma decisão discretizador pode necessitar ou não de todos os dados disponíveis. O discretizador é tido como local, quando utiliza uma informação parcial, no caso utiliza uma informação local em suas decisões, entretanto, o discretizador global utiliza todas as informações disponíveis para tomar suas decisões, porém, poucos discretizadores são locais (GARCIA et al., 2013).

Os métodos de discretização também podem ser agrupados em diretos ou incrementais. Os métodos diretos dividem o conjunto numérico pertencentes a uma variável contínuas em k intervalos iguais simultaneamente (*equal-Width*, *Equal-Frequency*), necessitando assim de um critério adicional para determinar o valor de k. Por outro lado, os métodos incrementais começam com uma simples discretização e ao decorrer do processo passa por um aperfeiçoamento para encontrar os melhores candidatos a fazerem parte da fronteira

de decisão, necessitando assim de um critério de parada, entretanto, apenas métodos supervisionados podem adotar o método incremental. Um típico processo de discretização consiste em quatro etapas:

- Ordenação do conjunto numérico pertencentes a uma variável contínuas a ser discretizada;
- Avalia os *cut-points* obtidos pelo método de discretização utilizado;
- Utiliza os *cut-points* para mapear o valor contínuo para o discreto;
- Finaliza a discretização.

3.4.1 Métodos Supervisionados

Tanto *Equal-Width* como *Equal-Frequency* se tratam de um método simples de discretização que requerem um parâmetro k que indica o número máximo de intervalos na discretização de uma variável A, onde k é um parâmetro fornecido pelo usuário. A melhor maneira para se obter o melhor valor de k é olhando o histograma e testando diferentes valores de k. Para *Equal-Width* o tamanho de cada intervalo para uma variável A é indicado utilizando a seguinte relação:

$$I_w = \frac{V^A_{max} - V^A_{min}}{k} \tag{3.14}$$

onde, I_w se trata do tamanho do intervalo, V^A_{max} e V^A_{min} são valor máximo e mínimo da variável A. Então, a partir do valor mínimo são criados os *points-cuts*, que separa os valores da variável A em k intervalos iguais, utilizando a relação:

$$CP_i = V^A_{min} + i \times I_w \tag{3.15}$$

sendo $i = 1, 2, \ldots, k-1$ e CP_i o enésimo cut-point. O *Equal-Frequency* é similar ao *Equal-Width*, porém, utiliza os valores dos N exemplos da variável A, e divide em k intervalos que apresentam aproximadamente o mesmo número de exemplos, tendo assim a seguinte relação:

$$i_f = \frac{N^A_u}{k} \tag{3.16}$$

onde, i_f se trata da frequência do intervalo e N^A_u a quantidade de exemplos da variável A. Apesar dos dois métodos serem sensíveis a escolha da variável k, o *Equal-Frequency*, por exemplo, apresenta um problema a mais, pois, algumas ocorrências de um valor numérico específico da variável A, podem ocorrer em diferentes intervalos, fazendo com que esse método apresente um difícil gerenciamento de suas fronteiras.

4 MODELO PROPOSTO

Nas seções seguintes são abordados tópicos sobre a arquitetura utilizada e detalhes sobre as representações e camadas que compõe o modelo. Dessa forma, a Seção 4.1 apresenta a arquitetura do modelo RAHE. A Seção 4.2 demostra definição formal do problema e algumas notações básicas. Na Seção 4.3 são apresentados os processos para se obter as representações visuais, textuais e do usuário. A Seção 4.4 aborda as camadas de atenção utilizadas para extrair o *embedding* e as multimodalidades. Na Seção 4.5 será definida a camada que prevê a popularidade. Por fim, na Seção 4.6 é definido o método de correção de erro que tem como objetivo minimizar o erro obtido pelo modelo RAHE.

4.1 REDE DE ATENÇÃO HIERÁRQUICA ESTENDIDA

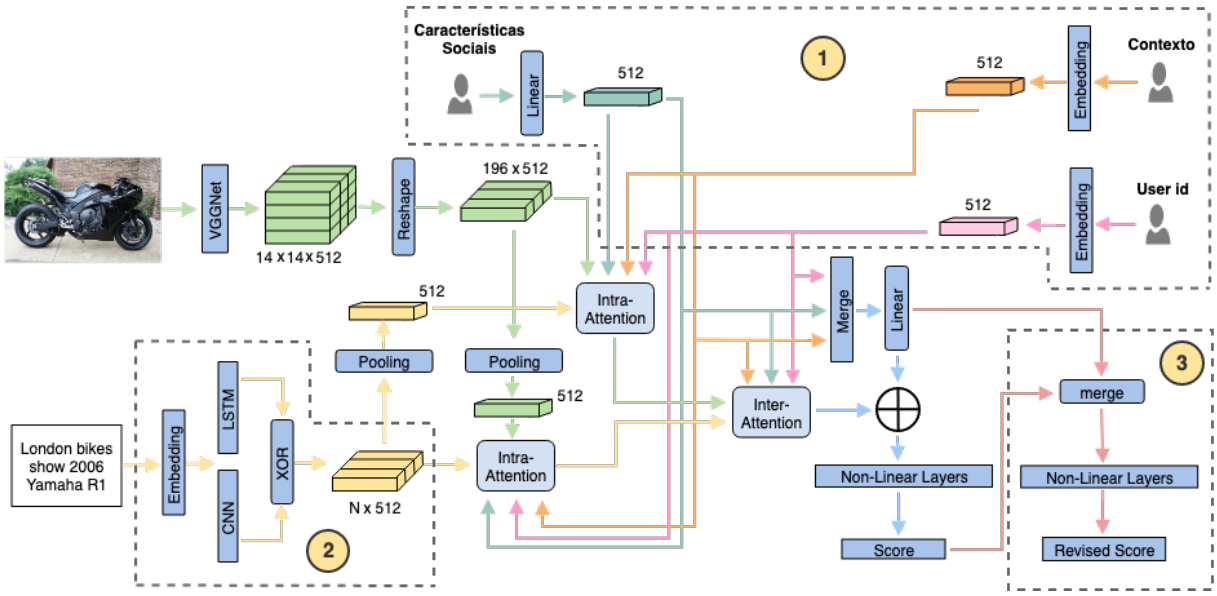


Figura 20 – Uma visão geral da arquitetura do RAHE. Sendo o bloco 1 as diferentes representações dos usuários, que podem ser obtidas pela informação de identificador único do usuário, informação sociais e informações contextuais. O bloco 2 representa os dois extratores de recursos textuais, o CNN e LSTM, que podem ser a nível de palavra ou caractere, onde o modelo utilizará apenas uma entre as duas arquiteturas, como indicado pelo módulo XOR. O bloco 3 apresenta o modulo de correção de erro que otimiza a saída do modelo RAHE a partir de uma rede de duas camadas *feedfoward*, obtendo assim a variação RAHE-CE.

Esse estudo utiliza arquitetura de Rede de Atenção Hierárquica guiada por Usuário (UHAN) desenvolvida por Zhang et al. (2018), que se trata de uma estrutura de multimodalidade, na qual, utiliza representações visuais e textuais guiadas por uma representação do usuário para tratar problema de predição de popularidade. Esse modelo apresenta

duas camadas de atenção, *intra-attention* e *inter-attention*, baseadas no modelo de atenção dupla, desenvolvido por Nam, Ha e Kim (2016), que atendem hierarquicamente as modalidades visuais e textuais.

Entretanto, o modelo UHAN apresenta limitações em relação à sua capacidade de generalização devido ao tipo de representação do usuário que compõe a sua arquitetura, uma representação que foca em usuários específicos. Em virtude disso, foi desenvolvida uma RAHE que realiza otimizações ao modelo UHAN a nível estrutural e a nível de recursos, como demostrado na Figura 20.

O RAHE foi criado para atender diferentes representações do usuário e representações textuais que incorporem características textuais a nível de caractere, com o objetivo de obter um modelo com diversidade e capacidade de generalização. Além disso, o modelo RAHE incorpora um método de correção de erro, que visa minimizar os erros obtidos na predição de popularidade, principalmente, os erros relacionados as amostras de popularidade de baixa frequência no conjunto amostral. Com a integração do método de correção de erro é obtido o modelo de Rede de Atenção Hierárquica Estendida com Correção de Erro (RAHE-CE).

4.2 DEFINIÇÃO DO PROBLEMA

Antes de dar continuidade sobre a estrutura do modelo RAHE, são introduzidas algumas notações importantes, assim como, a definição formal do problema de predição de popularidade, para melhorar compreensão do que será abordado nas demais seções. As matrizes são representadas por letras maiúsculas em negrito e os vetores apresentam uma notação com letras minúsculas em negrito. Em relação à popularidade, tem-se que, $\mathbf{C}$ se trata do conjunto de conteúdos gerados pelos usuários que contém N conteúdos diferentes. O conteúdo $\mathbf{c_i} \in \mathbf{C}$ é constituída de três informações básicas, a imagem, texto e dados do usuário, que apresentam, respectivamente, as seguintes notações $\mathbf{I}_i$, $\mathbf{D}_i$ e $\mathbf{u}_i$.

Logo, a partir das informações primárias são obtidas as suas representações visuais, textuais e de usuário, que apresentam, respectivamente a seguinte notação $\mathbf{V}_i$, $\mathbf{H}_i$ e $\mathbf{u}_i$. Uma vez que a popularidade de um conteúdo $\mathbf{C}_i$ gerado pelo usuário é dada por y_i. Tem-se que, baseado nas notações levantadas, a definição formal do problema de popularidade é dada por:

Problema: *Dada uma imagem* $\mathbf{I}_i$ *e um texto* $\mathbf{D}_i$ *que será publicado em uma mídia social pelo usuário* $\mathbf{u}_i$*, uma função* f *terá como objetivo prever a popularidade a partir de* $f(\mathbf{V}_i, \mathbf{H}_i, \mathbf{u}_i) \rightarrow y_i$*, onde seu erro de predição será corrigido por uma função* $\hat{f}(\mathbf{u}_i, y_i) \rightarrow \hat{y}_i$*.*

Para entender como são obtidas as representações textuais, visuais e do usuário, respectivamente, com as seguintes notações, $\mathbf{V}_i$, $\mathbf{H}_i$ e $\mathbf{u}_i$, foi utilizado um único conteúdo

$\mathbf{c}_i$, que contém as informações básicas $\mathbf{I}_i$, $\mathbf{D}_i$ e $\mathbf{u}_i$, para servir de suporte na introdução dessas representações utilizadas pelo modelo RAHE.

4.3 REPRESENTAÇÕES USADAS NO MODELO RAHE

Nessa seção será abordada como são obtidas as representações visuais, textuais e do usuário que serão dadas como entrada para o modelo RAHE.

4.3.1 Representação Visual

Tendo uma imagem $\mathbf{I}_i$ dada como informação visual, será aplicada a $\mathbf{I}_i$ uma função $E(\mathbf{I}_i)$ que irá extrair um vetor de características ν. Como visto na Seção 3.1, é possível usar um modelo treinado em um grande conjunto de dados e transferir seu conhecimento para um conjunto de dados menor. Então, foi utilizada uma VGGNet pré-treinada com a base da *ImageNet* (SIMONYAN; ZISSERMAN, 2014). Onde a função $E(\mathbf{I}_i)$ representa a VGG-19, na qual, será utilizada para extrair características de alto nível da imagem.

Entretanto, é necessário realizar alguns processamentos antes de inicializar a extração do vetor do espaço de características da imagem, pois algumas imagens apresentam escalas diferentes e é preciso uniformizar essas escalas. Por esse motivo, todas as imagens foram re-escaladas para uma escala de 448×448, como realizado em (ZHANG et al., 2018). Adotando a convenção dada por Nam, Ha e Kim (2016), que utiliza a última camada de *pooling* da VGGNet, é obtida um vetor do espaço de características da imagem com o seguinte formato $14 \times 14 \times 512$.

A quantidade de regiões que foram exploradas na imagem é indicada por 14×14, onde, cada uma dessas regiões contém um espaço de características igual a 512 que foram extraídos da imagem pela VGGNet. No total, existem 196 regiões que representam a informação visual dada pelo vetor $\mathbf{V}_i = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$ onde $m = 196$ e $\mathbf{v}_m \in \mathbb{R}^{512}$.

4.3.2 Representação Textual

Para cada imagem $\mathbf{I}_i$ existe uma descrição $\mathbf{D}_i = \{w_t\}_{t=1}^{l}$, sendo l o tamanho da descrição e w_t a palavra contida na descrição na posição t. Entretanto, foram extraídos dois tipos de representação textual, uma representação textual a nível de palavra e outra a nível de caractere. Os tópicos a seguir apresentam como cada uma dessas representações são extraídas e quais arquiteturas, presentes no estado da arte, são utilizadas como unidade extratora de características textuais.

4.3.2.1 Extração a nível de palavra

Cada descrição é dada por $\mathbf{D}_i$ que contém um tamanho l, na qual l deve seguir uma restrição lL, em que L é a quantidade máxima de palavras que são utilizadas para representar o texto. Cada palavra $\mathbf{w}_t$ pertencente a $\mathbf{D}_i$ será indexada. Em alguns casos em que $l < L$,

$L - l$ espaços de $\mathbf{D_i}$ são preenchidos com indexes nulos e em casos em que $l > L$, $l - L$ palavras são removidas do final de $\mathbf{D_i}$. Dessa forma é obtido um vetor $\mathbf{H}_i = [\mathbf{w}_1, \ldots, \mathbf{w}_L]$ que se trata da representação textual processada e indexada.

Como foi visto na Seção 3.2.1 é possível, através do *word embedding*, obter a partir de uma variável discreta uma representação vetorial de valores contínuos dentro de um espaço semântico, permitindo com que palavras similares fiquem próximas no espaço. Então, a partir de cada $\mathbf{w}_L$, através da Equação 3.4, será obtido o *embedding* $\mathbf{W_w} \in \mathbb{R}^{|V| \times 512}$, onde $|V|$ é o tamanho do vocabulário das palavras mapeadas. Transformando assim o vetor $\mathbf{H}_i$ no seguinte vetor $\hat{\mathbf{W}} = [\hat{\mathbf{w}}_1, \ldots, \hat{\mathbf{w}}_L]$, onde $\hat{\mathbf{w}}_L \in \mathbb{R}^{512}$.

Para a extração de características do *word embedding* da representação textual $\hat{\mathbf{W}}$, foram aplicados o LSTM, devido a apresentarem bons desempenhos em processamento de linguagem natural (TAI; SOCHER; MANNING, 2015), e o CNN, que se mostrou efetivo na classificação de sentenças (KIM, 2014). Para o LSTM, como visto na Seção 3.2, cada unidade da LSTM apresenta um *input-gate* $\mathbf{i}_t$, um *output gate* $\mathbf{o}_t$, um *cell-state* $\mathbf{c}_t$. Então, para cada $\hat{\mathbf{w}}_L \in \hat{\mathbf{W}}$ um correspondente $\mathbf{h}_t$ será calculado através das seguintes fórmulas.

$$\mathbf{i}_t = \sigma\left(\mathbf{W}^{i}_{\mathbf{w}}\hat{\mathbf{w}}_t + \mathbf{W}^{\mathbf{i}}_{\mathbf{H}}\hat{\mathbf{h}}_{t-1} + \mathbf{b}_i\right) \tag{4.1}$$

$$\mathbf{f}_t = \sigma\left(\mathbf{W}^{\mathbf{f}}_{\mathbf{w}}\hat{\mathbf{w}}_t + \mathbf{W}^{\mathbf{f}}_{\mathbf{H}}\hat{\mathbf{h}}_{t-1} + \mathbf{b}_f\right) \tag{4.2}$$

$$\mathbf{o}_t = \sigma\left(\mathbf{W}^{\mathbf{o}}_{\mathbf{w}}\hat{\mathbf{w}}_t + \mathbf{W}^{\mathbf{o}}_{\mathbf{H}}\hat{\mathbf{h}}_{t-1} + \mathbf{b}_o\right) \tag{4.3}$$

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tanh\left(\mathbf{W}^{\mathbf{c}}_{\mathbf{w}}\hat{\mathbf{w}}_t + \mathbf{W}^{\mathbf{c}}_{\mathbf{H}}\hat{\mathbf{h}}_{t-1} + \mathbf{b}_c\right) \tag{4.4}$$

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh\left(\mathbf{c}_t\right) \tag{4.5}$$

Onde $\circ$ é o produto de Hadamard e σ é a função de ativação sigmoide. Todos os $\mathbf{W_w}$, $\mathbf{W_H}$ e $\mathbf{b}$ se tratam de parâmetros que são aprendidos pela LSTM. Dessa forma é obtido $\hat{\mathbf{H}}^{\mathbf{w}}_{\mathbf{lstm}} = [\mathbf{h}_{\mathbf{w_1}}, \ldots, \mathbf{h}_{\mathbf{w_t}}]^{L}_{t=1}$, onde $\mathbf{h}_{\mathbf{w_t}} \in \mathbb{R}^{\mathbf{512}}$, gerando assim um *embedding* da representação textual $\hat{\mathbf{H}}^{\mathbf{w}}_{\mathbf{lstm}} \in \mathbb{R}^{L \times 512}$ a partir do LSTM.

Para o CNN, a operação de convolução sobre $\hat{\mathbf{W}}$, como visto na Seção 3.2, envolve um filtro $\phi \in \mathbb{R}^{h \times k}$ aplicado a uma janela h de *embeddings* $\hat{\mathbf{w}}_{\mathbf{L}}$ de k dimensões, gerando novas características textuais. Uma vez que $\hat{\mathbf{w}}_{\mathbf{L}} \in \mathbb{R}^{512}$, logo $k = 512$. Por exemplo, a característica $\mathbf{c}$ é gerada a partir de uma janela de *embeddings* $\hat{\mathbf{w}}_{L:L+h-1}$, com *stride* 1 , através de :

$$\mathbf{c_i} = f(\phi * \hat{\mathbf{w}}_{L:L+h-1} + \mathbf{b}) \tag{4.6}$$

Onde $\mathbf{b} \in \mathbb{R}$ que se trata do viés, f é um função não linear, podendo ser uma *ReLU* ou *Tanh.*A cada possível janela de *embeddings* $\{\mathbf{w}_{1:h}, \mathbf{w}_{2:h+1}, \ldots, \mathbf{w}_{L-h+1:L+h-1}\}$ é aplicado um filtro ϕ obtendo o seguinte *feature map*:

$$\mathbf{C} = [c_1, c_2, ..., c_{L-h+1}] \tag{4.7}$$

Onde $\mathbf{C} \in \mathbb{R}^{L-h+1}$. Em seguida é aplicado a operação de *max-pooling* (COLLOBERT et al., 2011):

$$c = \mathsf{max}\,(\mathbf{C}) \tag{4.8}$$

Dessa forma c é obtida como característica mais importante dentre todas existentes no *feature map* obtido pelo filtro. Entretanto, esse exemplo apresenta a extração apenas de uma única característica para apenas um único filtro. A ideia é utilizar múltiplos filtros com diferentes tamanhos de janela, obtendo assim, recursos textuais de n-grams a nível de palavras.

Tendo um conjunto de janelas $\{\omega_1, \omega_1, \ldots, \omega_\beta\}$, onde ω_β se trata da janela e β a quantidade de janelas, e um conjunto de filtros $\phi_\beta = \{\phi_{11}, \phi_{21}, \ldots, \phi_{n\beta}\}$, sendo $n\beta$ a quantidade de filtros e β a janela utilizada em cada filtro. Tem-se que, para cada filtro pertencente a uma janela β será aplicada as Equações 4.6 e 4.8, obtendo assim $\mathbf{C}_\beta = [c_{11}, c_{11}, c_{n\beta}]$, sendo $\mathbf{C}_\beta \in \mathbb{R}^{\rho \times n\beta}$, em que ρ representa o feature map. Então, ao agrupar todos os features maps obtidos para cada filtro, será gerado $\hat{\mathbf{C}} = [\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_\beta]$, na qual $\hat{\mathbf{C}} \in \mathbb{R}^{\rho \times (n\beta * \beta)}$.

Logo $\hat{\mathbf{C}}$ contém todas as características relevantes extraídas de $\hat{\mathbf{W}}$ que são passadas para uma camada de rede neural:

$$\mathbf{H}^{\mathbf{w}}_{\mathbf{cnn}} = \tanh\left(\mathbf{W}_{\mathbf{c}}\hat{\mathbf{C}} + \mathbf{B}\right) \tag{4.9}$$

onde $\mathbf{W}_{\mathbf{c}} \in \mathbb{R}^{\rho \times 512}$ e $\mathbf{B} \in \mathbb{R}^{\rho \times 512}$. Dessa forma é gerada o *embedding* da representação textual $\hat{\mathbf{H}}^{\mathbf{w}}_{\mathbf{cnn}} \in \mathbb{R}^{\rho \times 512}$ pela CNN.

4.3.2.2 Extração de palavras a nível de caracteres

Sendo $\mathbf{w}_t$ uma palavra contida em $\mathbf{D_i}$ na posição t e $[c_1, \ldots, c_l]$ uma sequência de caracteres de $\mathbf{w}_t$, tendo l como o tamanho de $\mathbf{w}_t$. Sendo ζ o vocabulário de indexes de caracteres, é possível, a partir da Equação 3.4, obter o *embedding* de caracteres $\mathbf{W}_c \in \mathbb{R}^{d \times |\zeta|}$, sendo d a dimensionalidade e $|\zeta|$ o tamanho do vocabulário de caracteres.

Tendo que a palavra $\mathbf{w}_t$ a nível de caractere é dada pela matriz $\mathbf{C}_{\mathbf{w}_t} \in \mathbb{R}^{d \times l}$, onde a j^{th} coluna de $\mathbf{C}_{\mathbf{w}_t}$ corresponde ao caractere *embedding* $\mathbf{c}_j$ que pertence a $\mathbf{c}^{th}$ coluna de $\mathbf{W}_c$. Então, para cada palavra $\mathbf{w}_t$ de $\mathbf{D}_i$ será obtido um $\mathbf{C}_{\mathbf{w}_t}$, gerando um vetor $\hat{\mathbf{D}}_i = [\mathbf{C}_{\mathbf{w}_1}, \mathbf{C}_{\mathbf{w}_2}, \ldots, \mathbf{C}_{\mathbf{w}_t}]$. Concatenando cada $\mathbf{C}_{\mathbf{w}_t}$ pertencente $\hat{\mathbf{D}}_i$ será obtido $\ddot{\mathbf{D}}_{\mathbf{i}} = [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n]$, onde n nada mais é do que a quantidade de caracteres contidos em

$\mathbf{D}_i$. Porém, n deve seguir uma restrição nN, em que N é a quantidade máxima de caracteres que são utilizadas para representar a informação textual, dessa forma, em casos em que $n < N$, $N-n$ espaços de $\ddot{\mathbf{D}}_i$ são preenchidos com *embeddings* nulos e em casos em que $n > N$, $n - N$ caracteres são removidas do final de $\ddot{\mathbf{D}}_i$. Obtendo assim a representação textual $\mathbf{H}_i = [\mathbf{c}_1, \ldots, \mathbf{c}_N]$.

Então, para extrair as características de $\mathbf{H}_i$ foi aplicado o LSTM e o CNN, como realizado na Seção 4.3.2.1. Logo, para cada $\mathbf{c}_N$ pertencente a $\mathbf{H}_i$ um correspondente $\mathbf{h}_N$ será calculado através das seguintes equações:

$$\mathbf{i}_N = \sigma\left(\mathbf{W}_{\mathbf{c}}^{\mathbf{i}}\hat{\mathbf{c}}_N + \mathbf{W}_{\mathbf{H}}^{i}\hat{\mathbf{h}}_{N-1} + \mathbf{b}_i\right) \tag{4.10}$$

$$\mathbf{f}_N = \sigma\left(\mathbf{W}_{\mathbf{c}}^{f}\hat{\mathbf{c}}_N + \mathbf{W}_{\mathbf{H}}^{f}\hat{\mathbf{h}}_{N-1} + \mathbf{b}_f\right) \tag{4.11}$$

$$\mathbf{o}_N = \sigma\left(\mathbf{W}_{\mathbf{c}}^{o}\hat{\mathbf{w}}_N + \mathbf{W}_{\mathbf{H}}^{o}\hat{\mathbf{h}}_{N-1} + \mathbf{b}_o\right) \tag{4.12}$$

$$\mathbf{c}_N = \mathbf{f}_N \circ \mathbf{c}_{N-1} + \mathbf{i}_N \circ \tanh\left(\mathbf{W}_{c}^{c}\hat{\mathbf{w}}_N + \mathbf{W}_{\mathbf{H}}^{c}\hat{\mathbf{h}}_{N-1} + \mathbf{b}_c\right) \tag{4.13}$$

$$\mathbf{h}_N = \mathbf{o}_N \circ \tanh\left(\mathbf{c}_N\right) \tag{4.14}$$

Onde $\circ$ é o produto de Hadamard e σ é a função de ativação sigmoide. Todos os $\mathbf{W}_c$, $\mathbf{W}_H$ e $\mathbf{b}$ se tratam de parâmetros da LSTM que são aprendidos. Dessa forma é obtido $\hat{\mathbf{H}}_{lstm}^{c} = [\mathbf{h}_{\mathbf{c}_1}, \ldots, \mathbf{h}_{\mathbf{c}_l}]$, onde $\mathbf{h}_{\mathbf{c}_l} \in \mathbb{R}^{512}$, gerando assim um *embedding* da representação textual a nível de caractere $\hat{\mathbf{H}}_{lstm}^{c} \in \mathbb{R}^{N\times 512}$ através do LSTM.

Para o CNN são utilizados os mesmo conceitos adotados para aplicar a convolução a nível de palavra, utilizando as equações 4.6, 4.7 e 4.15 obtendo um $\hat{\mathbf{H}}_{\mathbf{cnn}}^{\mathbf{c}} \in \mathbb{R}^{\rho\times 512}$.

4.3.3 Representação dos Usuários

Para cada imagem publicada sempre existirá um usuário u_i associado, pois, para uma imagem existir é necessário ter um usuário vinculado a ela. Nessa seção são apresentados as três diferentes representações do usuário que podem ser obtidas para uma imagem.

4.3.3.1 Representação específica por usuário

Esse tipo de representação associa um usuário a uma imagem a partir de um identificador único. Zhang et al. (2018) utilizou essa representação para guiar seu modelo a entender quais representações são mais importantes para um determinado usuário, a representação textual ou a visual. Dessa forma, obteve-se a matriz $\mathbf{W}_\mathbf{u} \in \mathbb{R}^{n\times 512}$ de *embeddings*, onde n representa a quantidade de usuários únicos, tendo assim cada usário o seguinte *embedding* $\hat{\mathbf{u}} \in \mathbb{R}^{512}$. Através do *embedding* dos usuários é possível obter características escondidas de cada usuário, tais como suas preferências.

4.3.3.2 Representação social

É possível criar uma nova representação a partir de características que descrevam o contexto social do perfil de cada usuário, como quantidade de seguidores, média de visualizações, quantidade de grupos e etc. Então, dado que $\mathbf{f}_u = [f_1, \ldots, f_n]$ se trata do conjunto de características sociais de um usuário u_i, onde n representa a quantidade de características sociais do perfil do usuário. Dado que $\mathbf{f}_u \in \mathbb{R}^n$, a representação social é obtida após a aplicação de uma camada de rede neural *feedforward* nesse conjunto de características:

$$\hat{\mathbf{f}}_\mathbf{u} = \tanh\left(\mathbf{W}_u \cdot \mathbf{f}_u + \mathbf{b}_u\right) \tag{4.15}$$

onde $\mathbf{W}_u \in \mathbb{R}^{n \times 512}$ e $\mathbf{b}_u \in \mathbb{R}^{512}$. Gerando assim o um *embedding* da representação do usuário $\hat{\mathbf{f}}_\mathbf{u} \in \mathbb{R}^{512}$.

4.3.3.3 Representação contextual

Usuários distintos apresentam características distintas, entretanto, se forem usuários distintos e influentes, irá existir uma característica comum entre eles e, da mesma forma, se forem usuários distintos e não tão influentes, também irá apresentar uma característica comum entre si. Ou seja, usuários distintos com um mesmo nível de influência apresentam um determinado contexto em comum. Portanto, será utilizada a ideia de contexto comum entre perfis para propor uma nova representação de usuário, sendo o contexto responsável por agrupar usuários com perfis sociais similares.

A ideia central de se ter um contexto é agrupar usuários distintos, com perfis sociais similares, possibilitando o compartilhamento de informações, que nesse caso se tratam de informação textuais e visuais, que irão auxiliar o modelo a entender que tipo de informações são mais relevantes para esse tipo de contexto em que os usuários estão contidos.

Então, para definir o contexto de um usuário $\mathbf{u}_i$ são utilizadas as características sociais representativas dos perfis. Dessa forma, sendo $\mathbf{f}_i = [f_1, \ldots, f_n]$ as características do usuário $\mathbf{u}_i$ e n a quantidade de variáveis representativas do perfil, têm-se que $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_i]$ o conjunto de todas as caracteristicas dos i usuários, onde $\mathbf{F} \in \mathbb{R}^{i \times n}$.

Será aplicado sobre $\mathbf{F}$ uma função τ que irá gerar um conjunto de grupos $\mathbf{G}$ a partir das combinações das n características contidas em $\mathbf{F}$ de cada usuário i, como demostrado a seguir:

$$\tau\left(\mathbf{F}\right) \Rightarrow \mathbf{G} \tag{4.16}$$

$$\mathbf{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_k\}$$

$$\mathbf{g}_k = (f_1^{z1}, f_2^{z2}, \ldots, f_n^{zn})$$

dessa forma, $\mathbf{g}_k$ é uma n-upla que se trata do k^{th} grupo gerado na combinação e $\mathbf{f}_n^{z_n}$ o z^{th} valor discreto da variável n de $\mathbf{F}$. A restrição $\mathbf{g}_k \notin \mathbf{G} - \mathbf{g}_k$ deve ser atendida.

Entretanto, há uma necessidade que as n características representativas dos perfis sejam discretas, pois, se forem características com valores contínuos, existirá uma infinidade de combinações que irão ser realizadas ao aplicar o agrupamento τ.

Logo, sendo $\mathbf{f}_n$ uma variável contínua, será aplicada sobre ela o método supervisionado de discretização *equal-width* abordado na Seção 1.3. Então, são utilizadas as equações 3.14 e 3.15 para realizar a discretização de $\mathbf{f}_n$, obtendo uma nova variável discreta $\mathbf{d}_n = \{[d_0, d_1], (d_1, d_2], \ldots, (d_{m-1}, d_m]\}$, onde d_0 e d_m são respectivamente o valor mínimo e máximo de $\mathbf{f}_n$, sendo $d_i < d_i + 1$ para $i = 0, 1, \ldots, m-1$ tendo $\{d_1, d_2, \ldots, d_{m-1}\}$ como o conjunto de *cut-points*.

Ao realizar esse mesmo procedimento para as demais variáveis contínuas pertencentes a $\mathbf{F}$ é obtido o seguinte conjunto $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n]$. Dessa forma, é aplicada a função τ em $\mathbf{D} \oplus (F - (f \in D))$ para serem obtidos o conjunto de grupos $\mathbf{G} \in \mathbb{R}^{k \times n}$. A quantidade de grupos k do conjunto $\mathbf{G}$ podem ser definidos a partir dos *cut-points* obtidos na discretização. Pois, $\mathbf{d}_n$ apresenta uma cardinalidade r que se trata da quantidade de *cut-points* obtidas na fase de discretização, que significa a quantidade de categorias geradas para a variável discreta. Logo, a quantidade total de grupos k pode ser definido a partir da seguinte equação:

$$k = \prod_{i=0}^{n} r_{\mathbf{d}_n} \tag{4.17}$$

onde $r_{\mathbf{d}_n}$ é a cardinalidade r de $\mathbf{D}$. Os grupos que foram gerados são os contextos obtidos a partir das características representativas dos perfis dos usuários, dados por $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_k\}$, como demostrado na Figura 21. Então, cada contexto $\mathbf{c}_k$ foi convertido a um *one-hot encoding*, pela Equação 3.2, e a partir da matriz $\mathbf{W}_c \in \mathbb{R}^{k \times 512}$ foi obtido o *embedding* contextual $\hat{\mathbf{c}}_k \in \mathbb{R}^{512}$, como no exemplo apresentado pela Figura 22.

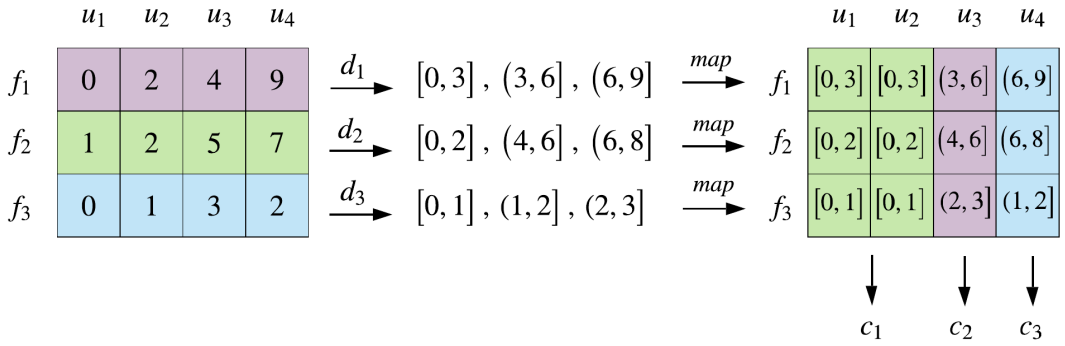


Figura 21 – Exemplo de como são obtidos os contextos dos usuários a partir das variáveis sociais representativas do perfil. $\mathbf{d}_1$, $\mathbf{d}_2$ e $\mathbf{d}_3$ são as discretizações das variáveis contínuas $\mathbf{f}_1$, $\mathbf{f}_2$ e $\mathbf{f}_3$, respectivamente. Onde essas discretizações são utilizadas para mapear os usuários para seus contextos $\mathbf{c}_1$, $\mathbf{c}_2$ e $\mathbf{c}_3$.

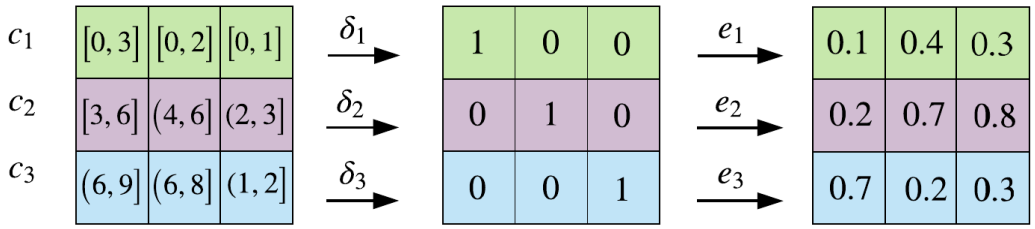


Figura 22 – Exemplo do *embedding* aprendido durante o treinamento para cada um dos contextos $\mathbf{c}_1$, $\mathbf{c}_2$ e $\mathbf{c}_3$.

Entretanto, nem todos os k contextos obtidos a partir da combinação realizada por τ apresentaram amostras que o representem durante a etapa de treinamento do modelo, gerando assim *embedding* contextuais nulos, pois $\mathbf{W}_c$ não irá aprender informações sobre esses contextos. Então, com intuito de evitar que usuários que serão avaliados pelo modelo estejam em um contexto que não foi aprendido durante o treinamento, foi definida a similaridade contextual.

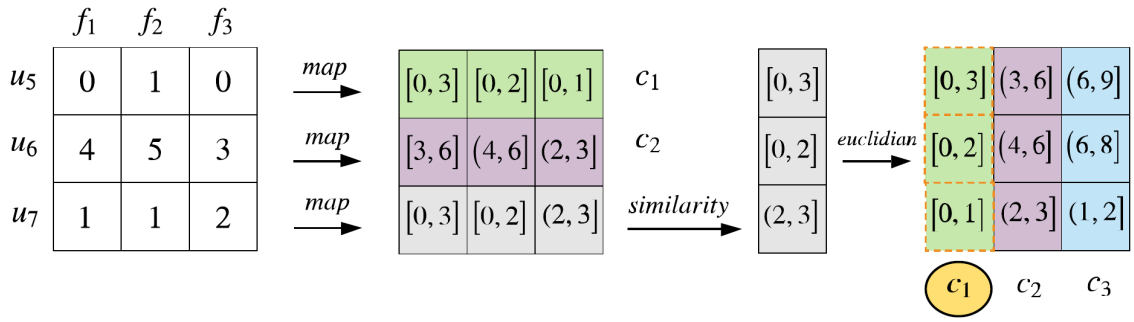


Figura 23 – Exemplo de um contexto que representa o usuário $\mathbf{u}_7$ que não foi aprendido durante a etapa de treinamento. Dessa forma, é realizado o cálculo da similaridade contextual entre o contexto que representa o usuário $\mathbf{u}_7$ e os demais contextos aprendidos durante a etapa de treinamento, no intuito de obter o contexto mais similar ao seu. A partir do cálculo da similaridade, o contexto $\mathbf{c}_1$ é selecionado como o contexto mais similar para o usuário $\mathbf{u}_i$.

A similaridade contextual tem como objetivo resgatar o contexto mais similar a um usuário que apresente um *embedding* contextual nulo. Então, para medir a similaridade entre contextos foi utilizada a distância euclidiana:

$$\mathbf{d}\left(\left(f_1^{c_k}, \ldots, f_n^{c_k}\right), \left(f_1^{c_j}, \ldots, f_n^{c_j}\right)\right) = \sqrt{\left(f_1^{c_k} - f_1^{c_j}\right)^2 + \cdots + \left(f_n^{c_k} - f_n^{c_j}\right)^2} \tag{4.18}$$

Onde, dado um contexto $\mathbf{c}_k$ e um conjunto de contextos com *embedding* não nulos C', será calculada a distância $\mathbf{c}_k$ para todos os contexto contidos em $\mathbf{C}'$ gerando assim um

vetor $\dot{\mathbf{d}} = [d_1, d_2, \ldots, d_j]$, onde d_j se trata da distância entre $\mathbf{c}_k$ e o contexto $\mathbf{c}_j$, como demostrado na Equação 4.18. Como se trata do contexto mais similar, será aplicada então $\hat{c} = min(\dot{\mathbf{d}})$, onde $\hat{c}$ é o contexto mais similar. Dessa forma será obtido para todo usuário $\mathbf{u}_i$ uma representação do usuário com um *embedding* contextual não nulo e similar ao seu contexto, como apresentado na Figura 23.

4.4 CAMADAS DE ATENÇÃO HIERÁRQUICA

Nessa seção serão abordadas as camadas *intra-attention* e *inter-attention*, que atendem hierarquicamente as modalidades visuais e textuais guiadas por diferentes representações do usuário.

4.4.1 *Intra-Attention*

A primeira etapa do modelo tem como objetivo obter um *embedding* representativo da informação visual e textual levando em conta a influência da representação utilizada para o usuário. Dessa forma, será dada para cada modalidade uma atenção diferente, onde o módulo de *intra-attention* irá utilizar a correlação entre o *embedding* do usuário, o *embedding* da modalidade que ser quer dar a atenção e o *embedding* da modalidade auxiliar. Antes de dar continuidade sobre essa camada, é necessário compreender quais as dimensões adotadas para as representações que são utilizadas como entrada do modelo.

A representação visual é dada por $\mathbf{V} \in \mathbb{R}^{196\times512}$ como apresentado na Seção 4.3.1. Já a representação textual depende muito do extrator de características textuais que será utilizado, pois, como abordado na Seção 4.3.2, será utilizada ou a CNN ou a LSTM como extrator e, além disso, deve ser levado em consideração o nível de representação textual que será empregue, se será nível de palavra ou de caractere. Então, a camada extrator, apresentada na Figura 20, irá fornecer uma saída unificada como $\hat{\mathbf{H}} \in \mathbb{R}^{N\times512}$, sendo N a quantidade de palavras ou caracteres utilizados, que irá representar uma dessas representações dadas pelo extrator:

$$\hat{\mathbf{H}}^{\mathbf{w}}_{\mathbf{lstm}} \in \mathbb{R}^{50\times512}$$

$$\hat{\mathbf{H}}^{\mathbf{w}}_{\mathbf{cnn}} \in \mathbb{R}^{\frac{50}{2}\times512}$$

$$\hat{\mathbf{H}}^{\mathbf{c}}_{\mathbf{lstm}} \in \mathbb{R}^{250\times512}$$

$$\hat{\mathbf{H}}^{\mathbf{c}}_{\mathbf{cnn}} \in \mathbb{R}^{\frac{250}{2}\times512}$$

Sendo $\mathbf{w}$ extração de características a nível de palavra, onde é utilizado um $N = 50$, convenção adotada por Zhang et al. (2018), e $\mathbf{c}$ a extração de características a nível de caractere apresentando um $N = 250$. Podem ser adotados outros valores para N a nível de palavras e caracteres, porém, para esse estudo não foi focada a otimização desse parâmetro. *lstm* e *cnn* representam o tipo de unidade extratora utilizada. Como a

representação textual, a representação de usuário também apresenta suas particularidades, pois, pode ser composta por um ou k tipos de representações do usuário que será dada por $\hat{\mathbf{u}} \in \mathbb{R}^{512}$. Sendo $\mathbf{u}$ uma única representação de um usuário e $\mathbf{U} = \mathbf{u}_1 \oplus \cdots \oplus \mathbf{u}_k$ as k tipos de representações do usuário agrupadas pela operação de concatenação, teremos que:

$$\hat{u} = \begin{cases} k = 1, & \mathbf{u} \\ k > 1, & \mathbf{W}_u\mathbf{U} + \mathbf{b} \end{cases} \tag{4.19}$$

Onde $\mathbf{W}_u \in \mathbb{R}^{(k*512)\times 512}$ e $\mathbf{b} \in \mathbb{R}^{512}$ compõe a função linear aplicada sobre o agrupamento de k tipos de representações do usuário. Então, são obtidos $\mathbf{V}$, $\hat{\mathbf{H}}$ e $\hat{\mathbf{u}}$, que se tratam, respectivamente, da representação visual, representação textual e representação do usuário, como fonte de informação do modelo.

4.4.1.1 Modalidade visual

São utilizadas a representação textual $\hat{\mathbf{u}}$ e a representação do usuário $\hat{\mathbf{u}}$ para calcular o *embedding* de atenção visual da correlação dessas três informações. Logo, a matriz textual $\hat{\mathbf{H}}$ é convertida para um vetor $\bar{\mathbf{h}}$ através da aplicação de um *mean-pooling*, dado pela equação 4.20:

$$\bar{\mathbf{v}} = mean\left(\hat{\mathbf{H}}_i\right) \tag{4.20}$$

Sendo $\bar{\mathbf{h}} \in \mathbb{R}^{512}$. Dessa forma, ambos $\bar{\mathbf{h}}$ e $\hat{\mathbf{u}}$ são vetores que apresentam uma dimensão $\mathbf{d} \in \mathbb{R}^{512}$. A formula que correlaciona as modalidades e obtém o *embedding* de atenção visual é dada por:

$$\mathbf{r}_{\mathbf{V},m} = \mathbf{W}^1_{\mathbf{V}}\left(\tanh\left(\mathbf{W}^1_{\mathbf{V_v}}\mathbf{v_m}\right) \circ \tanh\left(\mathbf{W}^{\mathbf{1}}_{\mathbf{V_u}}\hat{\mathbf{u}}\right) \circ \tanh\left(\mathbf{W}^{\mathbf{1}}_{\mathbf{V_t}}\bar{\mathbf{h}}\right)\right) \tag{4.21}$$

Onde $\mathbf{r}_{\mathbf{V},m}$ representa um *score* de importância da região m de uma imagem. $\mathbf{W}^1_{\mathbf{V}} \in \mathbb{R}^{1\times 512}$, $\mathbf{W}^1_{\mathbf{V}_t}$, $\mathbf{W}^1_{\mathbf{V}_u}$ e $\mathbf{W}^{\mathbf{1}}_{\mathbf{V_v}} \in \mathbb{R}^{512\times 512}$ são parâmetros que satisfazem o modelo e são aprendidos durante o treinamento. A interpretação intuitiva a partir da formula acima é que a modalidade visual tem sua relevância definida, para cada região da imagem, pela afinidade entre o texto e o usuário. Ou seja, a composição entre texto e usuário pode guiar o *embedding* de atenção visual e indicar qual região é mais importante para revelar a popularidade.

Então, supondo que $\alpha_{\mathbf{V}}$ se trata da distribuição de probabilidade da importância de cada região da imagem, que é dado por:

$$\alpha_{\mathbf{V}} = Softmax(r_{\mathbf{V}}) \tag{4.22}$$

baseado nessa distribuição de probabilidade, será obtida o *embedding* de atenção visual de todas as regiões da imagem dada por:

$$\dot{\mathbf{v}} = \sum_{m} \alpha_{\mathbf{V},m} \cdot \mathbf{v_m} \tag{4.23}$$

4.4.1.2 Modalidade textual

A modalidade textual se assemelha bastante ao que foi feito para visual, entretanto, se é utilizado a representação visual $\mathbf{V}$ como fator de correlação em conjunto com a representação do usuário $\hat{\mathbf{u}}$. Então, a representação visual é convertida em uma vetor, através o *mean-pooling*, que é aplicado em cada região da imagem, a partir da equação 4.24:

$$\bar{\mathbf{v}} = mean\left(\mathbf{V}_i\right) \tag{4.24}$$

obtendo assim $\bar{\mathbf{v}} \in \mathbb{R}^{512}$. Então, o *embedding* textual é obtido através da seguinte função:

$$\mathbf{r}_{\mathbf{T},t} = \mathbf{W}^1_{\mathbf{T}} \left(\tanh\left(\mathbf{W}^1_{\mathbf{T}_t}\mathbf{h}_t\right) \circ \tanh\left(\mathbf{W}^1_{\mathbf{T}_u}\hat{\mathbf{u}}\right) \circ \tanh\left(\mathbf{W}^1_{\mathbf{T}_\mathbf{v}}\bar{\mathbf{v}}\right)\right) \tag{4.25}$$

Onde $\mathbf{r}_{\mathbf{T},t}$ representa um score de importância de cada características $\mathbf{h}_t$. $\mathbf{W}^1_{\mathbf{T}} \in \mathbb{R}^{1\times512}$, $\mathbf{W}^1_{\mathbf{T}_t}$, $\mathbf{W}^1_{\mathbf{T}_u}$ e $\mathbf{W}^1_{\mathbf{T}_\mathbf{v}} \in \mathbb{R}^{512\times512}$ são parâmetros que satisfazem o modelo e são aprendidos durante o treinamento.

$$\alpha_{\mathbf{T}} = Softmax(\mathbf{r}_{\mathbf{T}}) \tag{4.26}$$

$$\dot{\mathbf{h}} = \sum_{m} \alpha_{\mathbf{T},t} \cdot \mathbf{h}_t \tag{4.27}$$

Sendo $\alpha_{\mathbf{T}}$ representação da distribuição de probabilidade da importância de todas as características do *embedding* textual e $\dot{\mathbf{h}}$ sendo o *embedding* de atenção textual. Em resumo, foram obtidos os *embedding* de atenção visual $\dot{\mathbf{v}}$ e textual $\dot{\mathbf{h}}$ que são utilizados na próxima camada do *inter-attention*.

4.4.2 *Inter-attention*

Essa camada captura as diferentes importâncias dos *embeddings* das modalidades visual $\dot{\mathbf{v}}$ e textual $\dot{\mathbf{h}}$ obtidos na Seção 4.4.1. A intuição por trás disso se dá através de que usuários tem concentrações diferentes em informações textuais e visuais em suas publicações. E um usuário pode focar mais em uma determinada modalidade para diferentes situações. Então, partindo dessa concepção foram definidos α_1 como a atenção da modalidade visual e α_2 como atenção da modalidade textual , tendo $\alpha_1 + \alpha_2 = 1$, dessa forma é obtida a

influência que cada modalidade apresenta para o cálculo da popularidade. A seguir são definidas as fórmulas para serem calculados o α_1 e α_2:

$$\mathbf{uv} = \mathbf{W}^2_{\mathbf{UVT}} \left(\tanh \left(\mathbf{W}^2_{\mathbf{V}} \dot{\mathbf{v}} \right) \circ \tanh \left(\mathbf{W}^2_{\mathbf{U}} \hat{\mathbf{u}} \right) \right) \tag{4.28}$$

$$\mathbf{uv} = \mathbf{W}^2_{\mathbf{UVT}} \left(\tanh \left(\mathbf{W}^2_{\mathbf{T}} \dot{\mathbf{h}} \right) \circ \tanh \left(\mathbf{W}^2_{\mathbf{U}} \hat{\mathbf{u}} \right) \right) \tag{4.29}$$

$$\alpha_1 = \frac{\exp\left(\mathbf{uv}\right)}{\exp\left(\mathbf{uv}\right) + \exp\left(\mathbf{ut}\right)} \tag{4.30}$$

$$\alpha_2 = \frac{\exp\left(\mathbf{ut}\right)}{\exp\left(\mathbf{uv}\right) + \exp\left(\mathbf{ut}\right)} \tag{4.31}$$

onde **uv** represente o score de relevância entre o usuário e a modalidade visual e **ut** represente o score de relevância entre o usuário e a modalidade textual. $\mathbf{W}^2_{\mathbf{UVT}} \in \mathbb{R}^{1 \times 512}$, $\mathbf{W}^2_{\mathbf{U}}$,$\mathbf{W}^2_{\mathbf{V}}$ e $\mathbf{W}^2_{\mathbf{T}} \in \mathbb{R}^{512 \times 512}$ são parâmetros que satisfazem o modelo e são aprendidos durante o treinamento. Então o *inter attention* é calculado a partir da seguinte equação:

$$\mathbf{s} = \alpha_1 \cdot \dot{\mathbf{v}} + \alpha_2 \cdot \dot{\mathbf{h}} \tag{4.32}$$

4.5 PREDIÇÃO DE POPULARIDADE

Para realizar a predição foi adicionado um módulo que leva em consideração o *embedding* do usuário com o *embedding* do *inter-attention*, dada pela seguinte formula:

$$\mathbf{s} := \mathbf{s} + \mathbf{W}^3_{\mathbf{U}} \hat{\mathbf{u}} \tag{4.33}$$

onde $\mathbf{W}^3_{\mathbf{U}} \in \mathbb{R}^{512X512}$. Em seguida são utilizadas duas redes neurais *feedfoward* para gerar a predição da popularidade. Tendo como equação final:

$$\hat{y} = \mathbf{W}^2_{\mathbf{F}} ReLU(\mathbf{W}^1_{\mathbf{F}} s + \mathbf{b}^1_{\mathbf{F}}) + {\mathbf{b}_{\mathbf{F}}}^2 \tag{4.34}$$

Onde ReLU represente a unidade linear retificada de ativação não linear. $\mathbf{W}^1_{\mathbf{F}} \in \mathbb{R}^{512 \times 512}$ e $\mathbf{b}^1_{\mathbf{F}} \in \mathbb{R}^{512}$ são os parâmetros da primeira camada. $\mathbf{W}^2_{\mathbf{F}} \in R^{512}$ e $\mathbf{b}^2_{\mathbf{F}} \in \mathbb{R}$ são os parâmetros da segunda camada. E $\hat{y}$ se trata da predição do score da popularidade. Foi utilizado o Erro Médio Quadrático como função de otimização.

4.6 CORREÇÃO DE ERRO

Frederick Gauss, com seus estudos sobre eventos da natureza, observou um comportamento padrão entre as amostras estudadas por ele, onde grande parte dos eventos ficam em torno de um valor médio com uma certa variabilidade. Esses eventos seguem uma

distribuição gaussiana, que se trata de uma curva simétrica em torno do seu ponto médio, apresentando assim seu famoso formato de sino.

$$f\left(x\right) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)} \tag{4.35}$$

Essa distribuição estatística, definida pela Equação 4.35, onde μ é a média e σ é a variância, apresenta uma área sob a curva que determina a probabilidade de ocorrer um determinado evento, como demostrado na Figura 24.

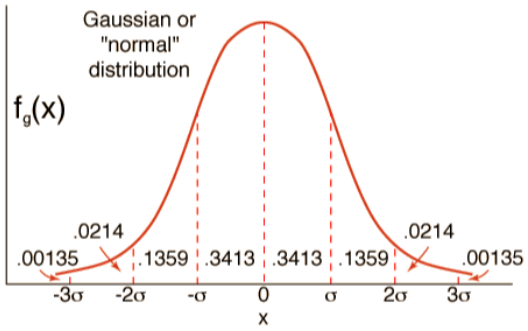


Figura 24 – O ponto mais alto da curva corresponde ao valor médio, e as extremidades correspondem aos valores que mais se afastam da média. .

Em problemas de predição popularidade, na maioria dos casos, as amostras dos valores de popularidade obedecem uma distribuição Gaussiana. Dessa forma, grande parte dos eventos e amostras se concentram em torno de um valor médio. Entretanto, valores de popularidade que se afastam desse valor médio apresentam probabilidade baixa de ocorrência, como demostrado na Figura 24.

Realizar a predição em cima de valores que não aparecem com muita frequência no conjunto amostral é um desafio enfrentado pelos modelos de regressão em aprendizagem de máquina, pois, como não existem amostras suficientes, o modelo não consegue aprender padrões específicos desses eventos. Portanto, a tendência é que o erro médio para esses eventos mais afastados do valor médio sejam maiores. Dessa forma, a predição para os valores que estão presentes na extremidade da curva da gaussiana são mais afetados, como o demostrado na Figura 25.

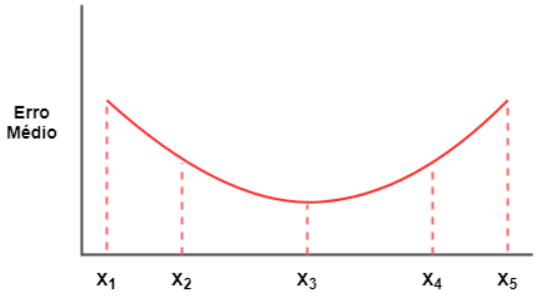


Figura 25 – Tendência do erro médio obtido sobre determinados valores de popularidade.

Em virtude disso, foi definido o método de correção de erro, que tem como principal objetivo aprender e minimizar os erros obtidos pela camada de predição de popularidade do modelo RAHE, principalmente, os erros relacionados as amostras de baixa frequência no conjunto amostral. Gerando assim uma Rede de Atenção Hierárquica Estendida com Correção de Erro (RAHE-CE). Esse módulo é integrado ao RAHE e utiliza apenas a informação $\hat{y}$ e a representação do usuário $\hat{\mathbf{u}}$ como entradas:

$$\hat{\mathbf{s}} := \hat{\mathbf{u}} + \hat{y} \tag{4.36}$$

Em seguida são utilizadas duas redes neurais *feedfoward* para gerar a predição da popularidade revisada, da mesma maneira da Seção 4.5. Tendo como equação final:

$$\bar{y} = \mathbf{W}_{\mathbf{F}}^{4} ReLU(\mathbf{W}_{\mathbf{F}}^{3} s + \mathbf{b}_{\mathbf{F}}^{3}) + \mathbf{b}_{\mathbf{F}}^{4} \tag{4.37}$$

Onde ReLU represente a unidade linear retificada de ativação não linear. $\mathbf{W}_{\mathbf{F}}^{3} \in \mathbb{R}^{512\times512}$ e $\mathbf{b}_{\mathbf{F}}^{3} \in R^{512}$ são os parâmetros da primeira camada. $\mathbf{W}_{\mathbf{F}}^{4} \in \mathbb{R}^{512}$ e $\mathbf{b}_{\mathbf{F}}^{4} \in \mathbb{R}$ são os parâmetros da segunda camada. Também foi utilizado o Erro Médio Quadrático como função de otimização. Gerando assim um $\bar{y}$ que tem como proposito, diminuir o erro médio, como demostrado na Figura 26.

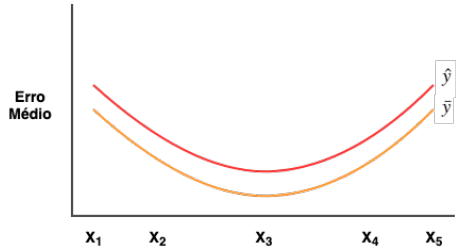


Figura 26 – Tendência do erro médio obtido sobre determinados valores de popularidade para o modelo RAHE *versus* RAHE-CE.

5 EXPERIMENTOS

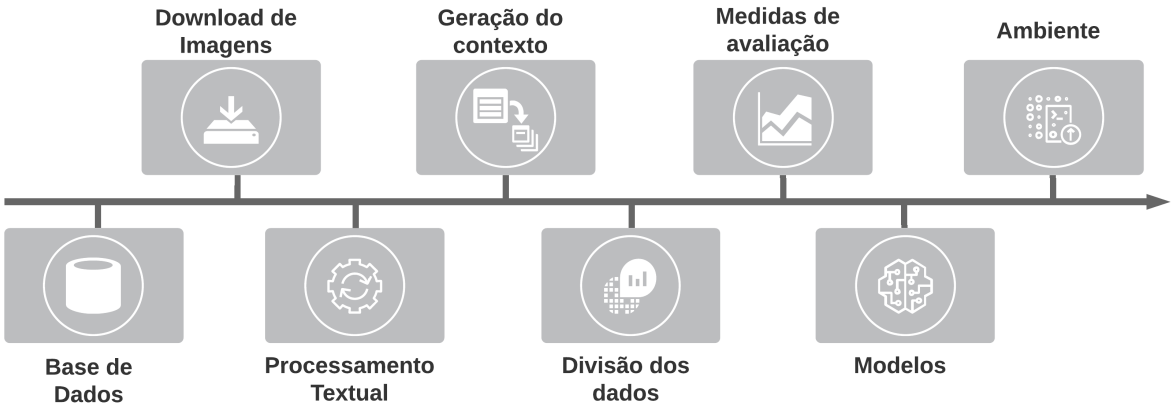


Figura 27 – O fluxo das etapas exploradas durante a preparação dos experimentos do modelo RAHE.

Essa seção descreve os detalhes de como foram conduzidos os experimentos. Na Seção 5.1 apresenta uma visão geral de como foi obtida a base de dados. A Seção 5.2 demostra todas as etapas dos processamentos realizados sobre a base para se obter as informações textuais, visuais e do usuário. A Seção 5.3 apresenta os três cenários em que os modelos são analisados. A Seção 5.4 introduz as métricas utilizadas na avaliação dos modelos. Na Seção 5.5 apresenta os *baselines* que são utilizados para confrontarem o modelo RAHE e suas variações. E por fim, a Seção 5.6 apresenta o ambiente utilizado durante os experimento e os parâmetros aplicados sobre a rede RAHE e suas variações.

5.1 BASE DE DADOS

A TPIC [1] se trata de uma base que contém $680k$ instâncias diferentes, onde cada instancia apresenta informações e metadados da imagem publicada e dados estruturados relacionados ao perfil. Essa base foi obtida a partir do Flickr durante um período de três anos, apresentando publicações entre os anos de 2001 e 2019. A Tabela 2 apresenta um resumo dos dados contidos nessa base.

O *score* presente na Tabela 2, se trata do valor de popularidade, referente a uma publicação, obtido a partir da função de *log-normalization*, como definido em (WU et al., 2017). Muitos artigos utilizam como popularidade a quantidade de *views* que uma publicação alcançou, porém, esse valor pode variar de zero a milhões. Isto é, utilizando apenas o valor da quantidade *views* como medida de popularidade, pode ocasionar grandes

[1] Base de dados pública utilizada por Wu et al. (2017) e disponibilizada na seguinte url: https://github.com/social-media-prediction/TPIC2017

Tabela 2 – Todos os metadados presentes na base de dados TPIC.

Features	**Descrição**
pid	O id único da imagem/publicação
uid	O id do usuário que realizou a publicação
commentcount	Quantidade de comentários recebidos na publicação
haspeople	Se a publicação apresenta pessoas marcadas
titlelen	Comprimento do título
deslen	Comprimento da descrição
tagcount	Quantidade de *tags* associadas a publicação
avgview	Média de visualizações que o usuário apresenta
groupcount	Quantidade de grupos que o usuário faz parte
avgmembercount	Média da quantidade de usuários pertencentes a aos grupos que o usuário faz parte
year	Ano em que a publicação foi realizada
hour	Hora em que a publicação foi realizada
minute	Minuto em que a publicação foi realizada
url	Url da imagem publicada
score	Popularidade da publicação

variações nos valores de popularidade. A função de *log-normalization* é utilizada com o objetivo de suprimir essas variações que diferentes publicações podem apresentar, gerando um novo valor que será tomado coma a popularidade obtida por uma publicação. A *log-normalization* é definida pela função da Equação 5.1.

$$score = log_2 \frac{r}{d} + 1 \tag{5.1}$$

onde *score* é a popularidade normalizada, r a quantidade de *views* e d a quantidade de dias que se passaram após a publicação.

5.2 PROCESSAMENTO DOS DADOS

Apesar da base de dados conter uma grande variedade de metadados relacionados a publicação, que se tratam de dados estruturados, ainda faltam as informações não estruturadas, como a imagem, título e descrição utilizados na publicação. Então, para coletar essas informações foi necessário construir um *Scrape* [2] em NodeJs [3] que tem como propósito resgatar os dados não estruturados com auxílio da biblioteca flickr-sdk [4].

[2] É uma técnica na qual um programa de computador extrai dados de uma saída legível por humanos provenientes de outro programa

[3] É um interpretador, com código aberto, de código JavaScript de modo assíncrono e orientado a eventos, focado em migrar a programação do JavaScript do lado do cliente para os servidores, criando assim aplicações de alta escalabilidade, capazes de manipular milhares de conexões/requisições simultâneas em tempo real, numa única máquina física. https://nodejs.org/en/

[4] É uma biblioteca baseada em superagente e todos os métodos que fazem chamadas de Application Programming Interface (API) retornarão uma instância de solicitação superagente configurada para a solicitação. Isso significa que você pode fazer qualquer coisa com as solicitações do Flickr que você pode fazer com o superagente

Após obter as descrições de cada imagem e realizar o *download* de todas as imagens e relacionadas as publicações contidas na base de dados, houve uma redução da base de dados de 680K instâncias para 587K. Isso se deve ao fato de muitos *links* utilizados para realizar o download das imagens estarem quebrados, impossibilitando assim o *download* da imagem e afetando diretamente na base, pois, não é do interesse da abordagem que está sendo adotada ter uma instancia na base que não contenha imagem associada a ela.

Após essa etapa de construção da base de dados, foram feitos alguns processamentos em cima das informações textuais associadas a cada publicação. A seguir são descritos os processamentos realizados, seguindo o processamento adotado por Zhang et al. (2018).

- Todas as informações textuais foram tokenizadas e colocadas em minúsculo;
- Foram removidos palavras das informações textuais que não estavam em inglês;
- A descrição e o título de cada instância foram unidas para formarem uma única fonte de informação textual;
- A partir das informações textuais foi gerado um vocabulário com a frequência de ocorrência cada palavra;
- Todas as palavras que apresentaram ocorrência menor que cinco foram removidas do vocabulário;
- Todas as instâncias que continham informação textual com menos de cinco palavras foram removidas do conjunto de dados;
- Foi gerado um vocabulário de caracteres contidos nas informações textuais.

Após esse processamento, a base de dados, que inicialmente continha $580K$ instâncias, foi reduzida para $256K$ instâncias, que apresentam informações visuais e textuais, que são manipuladas de acordo com os métodos definidos nas Seções 4.3.1 e 4.3.2 para gerarem representações visuais e textuais dadas como entradas para o modelo. Essa base também contém o identificador único do usuário, que se trata do *uid*, que servirá como representação especifica por usuário. A representação social é dada a partir das características centradas no perfil do usuário, que são o *avgview*, *groupcount* e *avgmembercount*. Porém, para se obter os contextos sociais comuns entre os usuários é necessário realizar uma série de procedimentos abordados na seção 4.3.3.3.

Devido *avgview*, *groupcount* e *avgmembercount* se tratarem de características que definem um contexto social, é possível obter grupos de contextos sociais comuns entre os usuários a partir dessas características. Como cada uma dessas características se tratam de variáveis contínuas, será aplicado o processo de discretização utilizando o algoritmo *equal-width*, Equações 3.14 e 3.15. Entretanto, é necessário definir o valor de k que indica o número máximo de intervalos na discretização de uma variável. Então, foi definido o

valor de k empiricamente, analisando o histograma obtido a partir de cada k avaliado. Dessa forma, foi levado em conta os valores de k que apresentaram um histograma mais uniforme, tendo a Tabela 3 um resumo do k escolhido para cada variável.

Tabela 3 – Resumo da quantidade de grupos obtidos para cada uma das variáveis contínuas que representam o perfil social do usuário.

	Variáveis Contínuas		
	avgview	**groupcount**	**avgmembercount**
k	60	200	2000
groups	38	8	21

Dessa forma, é aplicada a Equação 4.16 para obter um conjunto de grupos **G**, que nada mais são dos que os contextos sociais obtidos para os usuários. Gerando assim uma quantidade de 6384 contextos diferentes a partir da Equação 4.17, porém, apenas 200 contextos, dentre os 6384, apresentam amostras na base de dados que o representem. A Tabela 4 contém um resumo da base de dados após todo o processamento realizado.

Tabela 4 – Resumo geral das informações textuais, visuais e do usuário que foram extraídas da base de dados.

TPIC17				
# imagens	**# palavras**	**# caracteres**	**# usuários**	**# contextos**
256.337	121.458	79	341	200

A Figura 28 apresenta o histograma da popularidade das publicações em conjunto com a função de distribuição gaussiana, para se ter uma ideia de como está a distribuição da popularidade para esse conjunto de dados. Através da imagem é possível observar que 51% dos dados apresentam popularidade entre 5 e 7.5; 23% dos dados apresentam popularidade abaixo de 5; e 26% dos dados apresentam popularidade que estão acima de 7.5.

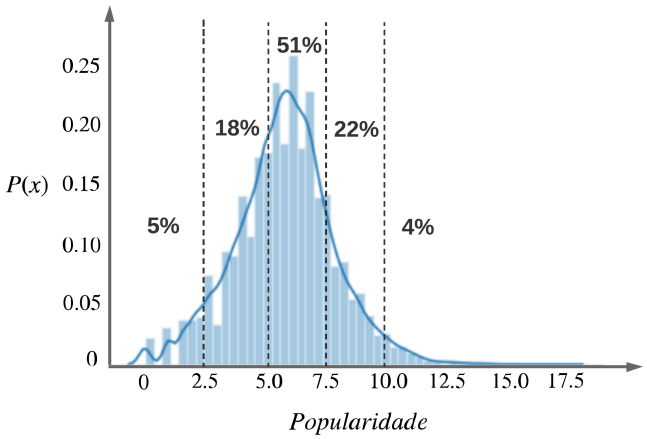


Figura 28 – Distribuição da frequência de valores de popularidade. .

5.3 DIVISÃO DOS CONJUNTOS DE DADOS

Para avaliação dos modelos são utilizados conjuntos de treinamento, validação e teste. Na qual, o conjunto de treinamento apresentará 70% dos dados e os demais 30% dos dados serão distribuídos para os conjuntos de validação e teste. Desses 30%, um terço é utilizado para validação, enquanto, dois terços são utilizados para teste. Essa divisão dos dados segue a abordagem adotada por Zhang et al. (2018). A Tabela 5 contém o resumo dessa divisão de dados.

Tabela 5 – Quantidade de instâncias para cada um dos conjuntos que serão utilizados durante os experimentos sobre os modelos.

Conjuntos		
Treinamento	**Validação**	**Teste**
179,43	25,633	51,267

Entretanto, alguns critérios tiveram que ser definidos para que esses dados fossem divididos da melhor maneira entre os conjuntos. Por exemplo, a divisão desses dados poderá conter usuários em comum entre as bases de treinamento, validação e teste? Zhang et al. (2018) adotou o critério de ordenar todos os dados pela data de publicação de maneira decrescente, ou seja, da publicação mais atual a publicação mais antiga, utilizando 70% dessas primeiras informações para o conjunto de treinamento e os 30% restantes seriam distribuídos de maneira aleatória para o conjunto de validação e teste. Entretanto, esse tipo de cenário criado apresenta uma composição de instâncias que avaliam ao mesmo tempo a capacidade de generalização e diversidade do modelo, podendo apresentar usuários em comum entre conjunto de treinamento, teste e validação, e apresentar usuários que só existem na base de treinamento, porém não existem no de validação e teste e vice-versa.

Porém, é necessário criar cenários que avaliem melhor o comportamento do modelo para as mais variadas situações em que ele pode ser confrontado. Com base nisso, foram definidos três cenários em que os modelos são treinados e avaliados. A partir desses cenários é possível avaliar a capacidade de generalização e diversidade dos modelos de maneira isolada, evitando assim, que resultados para determinadas condições sejam encobertos.

5.3.1 Cenário Disjunto

O cenário disjunto tem como a principal característica a ausência de usuários em comum entre as bases de treinamento e validação, assim como, entre treinamento e teste. No caso, esse tipo de cenário não apresenta interseção entre o conjunto de treinamento e os demais conjuntos. Esse tipo de abordagem é fundamental, uma vez que, se trata de um problema de predição de popularidade, nada mais justo do que ter um modelo que é capaz de prever usuários nunca vistos antes, dessa forma é possível medir a capacidade de generalização do modelo quando confrontado em tal situação.

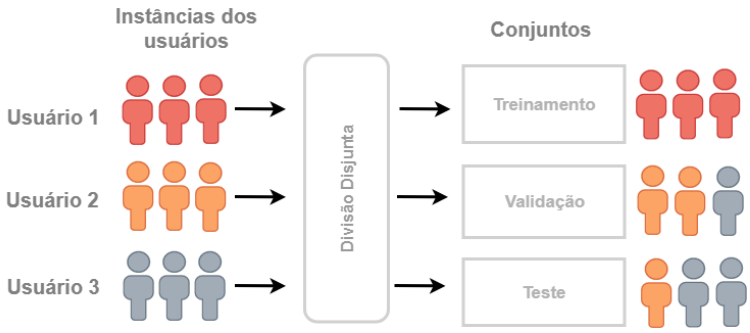


Figura 29 – Exemplo de como é realizada a divisão de dados para o cenário Disjunto.

A Figura 29 apresenta, de maneira mais prática como os usuários ficam dispostos nos conjuntos após a divisão, tendo para esse cenário instâncias do mesmo usuário presentes no conjunto de validação e teste, porém, não existe compartilhamento de instâncias de um usuário que está presente no conjunto de treinamento com os demais conjuntos.

A Tabela 6 contém um sumário do conjunto de treinamento obtido para o cenário disjunto, que apresenta um total 242 de usuários e 159 de contextos que compõe sua base, apresentando 99 usuários e 41 contextos de fora da base de treinamento.

5.3.2 Cenário Composto

O cenário composto realiza a separação da base de dados através da ordenação dos dados pela data de publicação, ordenando da publicação mais recente até a mais antiga, onde 70% dos dados serão utilizados para treinamento, na qual esses 70% são referentes a publicações mais recentes, tendo assim 30% para servir conjunto de validação e teste, que se tratam das últimas publicações. Nesse tipo cenário pode existir instâncias de um usuário compartilhadas entre treinamento, teste e validação, entretanto, nem todas as

Tabela 6 – Quantidade de usuários e contextos únicos presentes na base de treinamento do cenário Disjunto.

Cenário Disjunto		
	Usuários	**Contextos**
Treinamento	242	159
Dataset	341	200
	99	41

instâncias pertencentes a um usuário podem estar presentes em todas as bases, em alguns casos um usuário pode pertencer a base de teste e validação, mas não estar presente na de treinamento, assim como, estar presente na de validação e treino e não estar presente na de teste. A Figura 30 apresenta um exemplo de como os usuários são dispostos entre os conjuntos de treinamento, validação e teste.

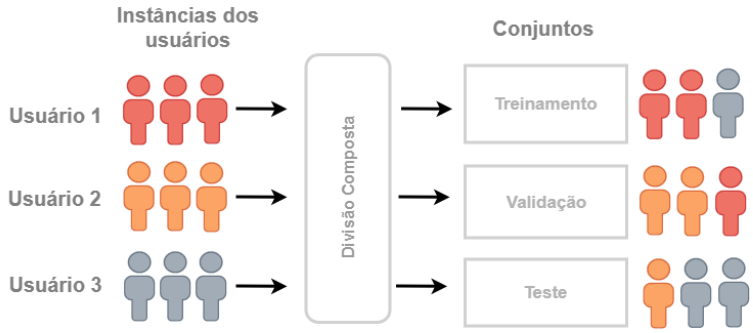


Figura 30 – Exemplo de como é realizada a divisão de dados para o cenário Composto.

Esse tipo de cenário é interessante para saber como o modelo se comporta em situações em que lida com alguns usuários que se conhece a priori e com usuários que nunca foram vistos antes pelo modelo, lidando ao mesmo tempo com diversidade e capacidade de generalização do modelo. A Tabela 8 contém um sumário do conjunto de treinamento para o cenário composto, que apresenta um total 316 de usuários e 190 de contextos que compõe sua base, tendo apresentando 25 usuários e 10 contextos de fora da base de treinamento.

5.3.3 Cenário Estratificado

O cenário estratificado apresenta uma divisão estratificada dos dados. Ou seja, irá conter nos conjuntos de treinamento, teste e validação, uma quantidade proporcional de instâncias de cada um dos usuários existentes na base, dessa forma, todas as bases irão apresentar pelo menos uma única instância daquele usuário. A Figura 31 demostra como é realizada a divisão das instâncias de cada usuário para cada base, tendo assim, as bases obtendo pelo menos um único exemplo de cada usuário em sua base de dados.

Tabela 7 – Quantidade de usuários e contextos únicos presentes na base de treinamento do cenário Composto.

Cenário Composto		
	Usuários	**Contextos**
Treinamento	316	190
Dataset	341	200
	25	10

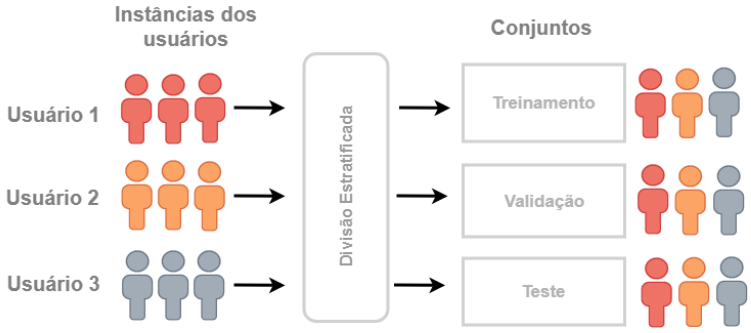


Figura 31 – Exemplo de como é realizada a divisão de dados para o cenário Estratificado.

Com esse tipo de cenário é possível analisar a diversidade do modelo, pois, como há uma informação a priori sobre o usuário no conjunto de treinamento, quando uma nova instancia do mesmo usuário aparecer no conjunto de validação e teste, o modelo terá uma maior facilidade em prever sua popularidade, uma vez que já se teve uma informação previa sobre o perfil desse usuário. A Tabela 8 contém um sumário do conjunto de treinamento para o cenário estratificado, que apresenta um total 341 de usuários e 200 de contextos que compõe sua base, não apresentando nenhum usuário ausente no conjunto de treinamento.

Tabela 8 – Quantidade de usuários e contextos únicos presentes na base de treinamento do cenário Estratificado

Cenário Estratificado		
	Usuários	**Contextos**
Treinamento	341	200
Dataset	341	200
	0	0

5.4 MEDIDAS DE AVALIAÇÃO

A qualidade de um modelo de regressão é dada pelo quão bem suas previsões correspondem ao valor real. O objetivo de um modelo de regressão é obter um erro residual mínimo, que é a diferença entre o valor real e o valor previsto. Existem várias medidas que calculam o erro residual e são largamente utilizadas na literatura para avaliar modelos de regressão. Para tratar o problema de predição de popularidade existem duas medidas largamente adotadas, o Erro Médio Quadrático (MSE) e o Erro Médio Absoluto (MAE) (WU et al., 2017).

A MAE é uma das medidas mais simples de entender das que são utilizadas para avaliar modelos de regressão. Ela calcula erro residual a partir do valor absoluto obtido entre valor real e o previsto divido pela média de todos os resíduos, como demostrado na Equação 5.2, sendo o valor absoluto utilizado para evitar que resíduos negativos e positivos sejam cancelados.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.2}$$

Como são calculados os valores absolutos residuais, cada resíduo contribui proporcionalmente para a quantidade total de erro, o que significa que erros maiores contribuirão linearmente para o erro global, ou seja, erros residuais mais discrepantes acabam por passar despercebidos. O Erro Médio Quadrático é equivalente ao MAE, porém, ao invés de utilizar o valor absoluto, utilizar o valor quadrático do erro residual, como demostrado na Equação 5.3.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5.3}$$

Como está sendo utilizado o quadrado da diferença, é comum que o MSE seja maior que o MAE. Embora cada resíduo no MAE contribua proporcionalmente ao erro total, o erro aumenta de forma quadrática no MSE. Isso significa que os erros residuais maiores contribuirão para um erro total muito maior no MSE do que no MAE. Da mesma forma, o modelo será mais penalizado por fazer previsões que diferem muito do valor real correspondente. Isso quer dizer que grandes diferenças entre o real e o previsto são punidas mais no MSE do que no MAE.

5.5 AVALIAÇÃO DOS MODELOS

Como são utilizadas representações visuais, textuais e do usuário nesse estudo, foram implementados alguns modelos no estado da arte que que podem ser aplicados sobre dados multimodais e ao problema de predição de popularidade. Dessa forma, é possível comparar a performance do modelo RAHE e suas variações com os demais modelos no

estado da arte. A seguir são levantados todos os modelos que foram implementados que serviram de parâmetro para avaliar o desempenho do modelo RAHE.

Multilayer Perceptron (MLP): Perceptron de Multicamadas é uma rede neural com uma ou mais camadas ocultas e com um número indeterminado de neurônios. A partir de um MLP de duas camadas ocultas utilizando uma função de ativação não Linear ReLU, seguindo uma estrutura parecida com a definida na Equação 4.34, serão gerados quatro modelos que utilizaram todos os tipos de representação do usuário para prever a popularidade. Os modelos MLP_{id} , MLP_{scl}, MLP_{ctt} e MLP_{ctt}^{sim} preveem a popularidade do usuário a partir das respectivas representações do usuário: representação por usuário específico; representação a partir de característica social; representação contextual e representação contextual por similaridade.

Concolutional Neural Netowork (CNN-VGG) (SIMONYAN; ZISSERMAN, 2014): O modelo CNN-VGG utilizará uma VGG-19 pré-treinada com a base da *ImageNet*, que atua sobre as imagens presentes nos conteúdos gerados pelos usuários, extraindo um vetor do espaço de características para prever sua popularidade.

Long Short-Term Memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997): Uma vez que que esse tipo de abordagem tem apresentado bons desempenhos em processamento de linguagem natural tai-etal-2015-improved, foi utilizada a representação textual de cada um dos conteúdos para auxiliar na predição da popularidade da LSTM.

Deep Multi-modal (LYNCH; ARYAFAR; ATTENBERG, 2015): Como se trata de uma abordagem largamente utilizada na literatura para predição de popularidade Hidayati:2017:PMI:3123266.3127903, (ANTOL et al., 2015) e (KARPATHY; LI, 2014), foi implementada utilizando as representações textuais e visuais extraídas do conteúdo gerado pelo usuário.

Dual-Attention (NAM; HA; KIM, 2016): Devido ao sucesso das abordagens de multimodalidades com mecanismo de atenção (ANTOL et al., 2015) e (KARPATHY; LI, 2014), alguns estudos de predição de popularidade começaram a adotar esse abordagem (ZHANG et al., 2018) e Wang:2019:FFA:3318299.3318305. Então, esse modelo foi implementado utilizando representações textuais e visuais com a finalidade de capturar as melhores representações entre essas duas modalidades para prever a popularidade.

UHAN (ZHANG et al., 2018): Modelo de multimodalidade que utiliza representações visuais e representações textuais guiadas por uma representação do usuário. Dessa forma esse é capaz de definir quais modalidades, textuais e visuais, são mais relevantes para um específico usuário. É o modelo de referência na maioria dos experimentos.

A seguir são definidas todas as variações do modelo RAHE para explorar todas as representações dos usuários, as características textuais de nível superior de palavra ou caractere exploradas pelas arquiteturas da LSTM e CNN e o método de correção de erro.

RAHE: Utiliza a representação textual , representação visual e representação do usuário e suas combinações para prever a popularidade do usuário. Essa versão utiliza apenas a LSTM como extrator de características textuais a nível de palavra.

$\boldsymbol{RAHE}_{arc}^{i_t}$: Sendo arc a arquitetura utilizada para extrair características a partir de uma representação textual e i_t o tipo de informação textual que será utilizada. Esse modelo utiliza a representação textual, representação visual e representação do usuário para prever a popularidade. Entretanto, tem como principal objetivo avaliar as contribuições de cada umas das representações textuais de que podem ser exploradas pelas arquiteturas da LSTM e CNN. Serão exploradas para cada um dos tipos de extrator, representações textuais a nível de palavra e caractere.

RAHE-CE : Também utiliza a representação textual , representação visual e representação do usuário para a predição de popularidade. Porém, é integrado o método de correção de erro, que tem como objetivo reduzir erro de popularidade obtido pelo modelo RAHE.

5.6 AMBIENTE

Para executar os experimentos sobre os modelos definidos na Seção 5.5, foi utilizada a plataforma em nuvem da Google (GCP) [5], principalmente o serviço da *Compute Engine*. Com o Compute *Engine*, é possível realizar a criação de máquinas virtuais (VMs) na infraestrutura do Google, ou instâncias maiores que executam Debian, Windows ou outras imagens padrão. Dessa forma foi utilizada a infraestrutura da Google para criar servidores com total autonomia de administração. A GCP disponibiliza o nível gratuito como uma oportunidade de aprender a usá-la sem custos, com credito de US$ 300. A partir do *compute engine* foram criadas três VMs que apresentam as mesmas configurações descritas na Tabela 9.

A biblioteca do Keras [6] foi instalada nesse ambiente e utilizada na implementação do modelo RAHE. Algumas configurações também foram pré-definidas para as representações e extratores de características que serão utilizadas no modelo RAHE, Tabela 10. Essa tabela apresenta a melhor configuração, para a arquitetura proposta, dentre as várias configurações testadas durante a etapa experimental.

[5] Uma suíte de computação em nuvem oferecida pelo Google, apresenta um conjunto de ferramentas de gerenciamento, fornecem uma série de serviços incluindo, computação, armazenamento de dados, análise de dados e aprendizagem de máquina.

[6] é uma biblioteca de rede neural de código aberto escrita em Python. Projetada para permitir a experimentação rápida com redes neurais profundas, ele se concentra em ser fácil de usar, modular e extensível.

Tabela 9 – Configurações das máquinas virtuais criadas para executarem os modelos.

Imagem	Deep Learning Image: PyTorch 1.1.0 and fastai m32
CUDA	CUDA 10.0
Biblioteca de Kernel Matemático	Intel® MKL-DNN, Intel® MKL
GPU	NVIDIA P100
Núcleos	16
Memória	60GB
Armazenamento	500GB SSD

Tabela 10 – Configurações pré-definidas das representações e dos extratores de características.

Tam. da representação textual a nível de palavra	50
Tam. da representação textual a nível de caractere	250
Dim. das representações dos usuários	512
Dim. da representação visual	192×512
Filtros da CNN para representação textual a nível de palavra	$3, 4, 5$
Quant. de filtros da CNN na representação textual a nível de palavra	256
Filtros da CNN para representação textual a nível de caractere	$10, 7, 5, 3$
Quant. de filtros da CNN na representação textual a nível de caractere	256
Quant. de estados escondidos da LSTM	512

Os modelos foram executados adotando o Adam como algoritmo de otimização, utilizando *batches* de 128 instâncias. Após o processamento de 64 *batches* o modelo utiliza o conjunto de validação para avaliar o seu desempenho, como abordado por Zhang et al. (2018). Se o modelo não apresentar, após 40 interações, nenhuma melhoria no resultado, há uma interrupção prévia no treinamento, onde, o melhor modelo avaliado pelo conjunto de validação é utilizado para avaliar o conjunto de teste.

6 RESULTADOS E DISCUSSÕES

Nesse capítulo são apresentadas as análises realizadas sobre o modelo RAHE e suas variações para os três cenários definidos na Seção 5.3. A Seção 6.1 apresenta a comparação do desempenho da versão final do modelo RAHE em relação aos *baselines*. Na Seção 6.2 foi realizada uma análise da contribuição individual das características sociais, visuais e textual. Também foi observada a contribuição da combinação entre modalidades visuais e textuais na Seção 6.3. Na Seção 6.4, foi realizada a comparação do modelo RAHE, utilizando seus vários tipos de representação de usuário, com os demais modelos. Avaliou-se a relevância das representações textuais para tratar o problema de popularidade na Seção 6.5. Na Seção 6.6, foi analisado os desempenhos apresentados pela modelo RAHE-CE. Foi avaliado quais informações podem ser extraídas a partir da visualização do *embedding* contextual obtido durante o treinamento na Seção 6.7. E por fim, na Seção 6.8 será analisado o comportamento da curva obtida pelo método de correção de erro.

6.1 AVALIAÇÃO DO MODELO RAEH

A Tabela 11 apresenta a comparação do desempenho da versão final do modelo RAHE em relação aos demais modelos do estado da arte que tratam o problema de popularidade. A versão final do modelo RAHE é composto pelas características contextuais por similaridade, representação textual a nível de caractere, com a CNN como unidade extratora de características, e o método de correção de erro.

Os resultados demostram que a abordagem adotada nesse estudo supera as demais abordagens em todos os cenários, evidenciando a capacidade de generalização e diversidade do modelo RAHE. Ao comparar o modelo *UHAN*, que se trata do *hard baseline*, com o modelo RAHE, são apresentadas melhorias de 48.14% no MSE e 20.45% no MAE, para o cenário Disjunto, de 18.84% no MSE e 9.21% no MAE, para o cenário Composto que foi introduzido pela abordagem *UHAN* de Zhang et al. (2018), e de 8.13% no MSE e 4.24% no MAE, para o cenário Estratificado. Nas próximas seções serão cobertos cada um dos estudos realizados para se obter a versão final do modelo RAEH.

Tabela 11 – Comparação entre os *baselines* e a versão final do modelo *RAHE*.

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
MLP_{ID}	4.026	1.542	2.733	1.272	2.106	1.104
MLP_{SCL}	2.922	1.335	2.619	1.245	2.876	1.298
MLP_{CTT}	3.056	1.323	2.564	1.220	2.180	1.129
$MLP_{CTT_{sim}}$	2.682	1.277	2.542	1.214	2.180	1.129
LSTM	3.377	1.420	3.393	1.427	1.749	0.976
CNN-VGG	3.963	1.538	3.759	1.489	3.404	1.412
Multi-Modal	3.358	1.440	3.436	1.438	1.657	0.954
Dual-Attention	3.105	1.391	3.051	1.324	1.631	1.645
UHAN	3.474	1.437	2.630	1.256	1.170	0.810
RAHE	**2.345**	**1.193**	**2.213**	**1.150**	**1.082**	**0.777**

6.2 CONTRIBUIÇÃO INDIVIDUAL DAS REPRESENTAÇÕES

A Tabela 12 contém os resultados obtidos para análise das contribuições individuais das representações textuais, visuais e do usuário avaliação. A partir desses resultados pode-se observar que o modelo que utiliza a característica contextual por similaridade $MLP_{CTT_{sim}}$, apresenta um melhor desempenho, em relação aos demais modelos, para os cenários Disjunto e Composto. Entretanto, seu desempenho para o cenário *Composto*, não difere tanto do resultado obtido para modelo que utiliza apenas a característica contextual do perfil MLP_{CTT}. Para o cenário Estratificado o modelo *LSTM*, que utiliza a representação textual, apresentou um desempenho expressivo em relação aos demais modelos.

Tabela 12 – Desempenho das análises individuais das representações visuais, textuais e do usuário

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
MLP_{ID}	4.026	1.542	2.733	1.272	2.106	1.104
MLP_{SCL}	2.922	1.335	2.619	1.245	2.876	1.298
MLP_{CTT}	3.056	1.323	2.564	1.220	2.180	1.129
$MLP_{CTT_{sim}}$	**2.682**	**1.277**	**2.542**	**1.214**	2.180	1.129
LSTM	3.377	1.420	3.393	1.427	**1.749**	**0.976**
CNN-VGG	3.963	1.538	3.759	1.489	3.404	1.412

Ao comparar o desempenho entre o modelo MLP_{CTT} e o MLP_{SCL}, que utiliza características sociais do perfil, é possível notar que, MLP_{CTT} apresentou desempenhos superiores nos cenários Composto e, principalmente, no Estratificado , entretanto, para

o cenário Disjunto apresentou um desempenho inferior devido a apresentar *embeddings* nulos para representar os usuários . Do mesmo modo, foi realizada a comparação entre $MLP_{CTT_{sim}}$ e o MLP_{SCL}, onde o $MLP_{CTT_{sim}}$ apresentou melhores desempenhos em todos os cenários, demostrando que o uso de contextos similares complementa a abordagem contextual adotada.

Essa análise revela que a abordagem adotada de gerar grupos de contextos compartilhados entre os usuários conseguiu superar os desempenhos obtidos pelas características sociais dos perfis, que já são bem difundidas no cenário de predição de popularidade. Esse método apresentou uma maior capacidade de generalização e de diversidade, mesmo os contextos compartilhados sendo obtidos a partir das características sociais dos perfis. Enquanto o modelo MLP_{SCL} apresenta dificuldades ao atuar sobre a diversidade dos dados no cenário Estratificado, por outro lado, a abordagem contextual atua bem sobre a diversidade dos dados, apresentando uma melhoria de 31.92% na MSE e 14.96% na MAE, em relação a MLP_{SCL} . Já para os demais cenários, a abordagem contextual apresenta melhorias, em relação MLP_{SCL} de 8.94% na MSE e 4.54% na MAE, para o cenário Disjunto, e 3.02% na MSE e 2.55% na MAE para o cenário *Composto* .

O modelo MLP_{ID} avalia a popularidade para um usuário específico, então, de certa forma, lida melhor com a diversidade, e isso é comprovado ao comparar seu desempenho no cenário Estratificado com os obtidos para a abordagem contextual. Porém, para uma abordagem que tem como propósito maior definir a popularidade de usuários específicos, seu desempenho não é substancial. Além disso, o modelo MLP_{ID} necessita de todos os usuários presentes na base, que são 341, enquanto, o método contextual precisa apenas de 200 contextos, na qual, esses 341 usuários estão imersos, Tabela 5 na Seção 5.3, reduzindo a complexidade do sistema.

Nos demais cenários é possível notar a limitação do modelo MLP_{ID}, onde apresenta resultados muito abaixo do modelo contextual, principalmente no cenário *Disjunto* , que apresentou um resultado abaixo do obtido do modelo *CNN-VGG* que, como levantado em estudos anteriores, esse tipo de abordagem apresenta uma maior dificuldade para extrair dados que correlacionem com a popularidade.

Apesar do modelo *LSTM* não apresentar desempenhos consideráveis nos cenários Disjunto e Composto, apresentando pouca capacidade de generalização, seu desempenho é perceptível quando analisado no cenário Estratificado. Onde é demostrada a capacidade que a representação textual apresenta na diversidade de um modelo. Ao comparar o modelo *LSTM* com o modelo MLP_{ID}, que apresentou o segundo melhor desempenho nesse cenário, o *LSTM* apresentou uma melhoria de 20.41% na MSE e 13.11% na MAE. O que indica a importância da aplicação de uma representação textual para tratar a popularidade de um conteúdo gerado pelo usuário.

6.3 COMBINAÇÃO ENTRE MODALIDADES VISUAIS E TEXTUAIS

A partir do desempenho apresentado pelo uso de representação textual para o cenário Estratificado, foi realizada uma análise sobre complementaridade entre a representação textual e visual. Ao analisar a Tabela 13, é possível notar que o uso de multimodalidades pelo modelo *Multi-Modal*, apresenta uma pequena melhoria no cenário Disjunto, quando comparado com a *LSTM*. Porém, para o cenário Composto, houve um desempenho inferior. Em relação ao cenário Estratificado, conseguiu apresentar um desempenho superior ao do *LSTM*, demostrando que a combinação entre as representações pode auxiliar no desempenho do modelo.

Tabela 13 – Resultados da combinação das representações utilizando multimodalidades e mecanismos de atenção.

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
LSTM	3.377	1.420	3.393	1.427	1.749	0.976
CNN-VGG	3.963	1.538	3.759	1.489	3.404	1.412
Multi-Modal	3.358	1.440	3.436	1.438	1.657	0.954
Dual-Attention	**3.105**	**1.391**	**3.051**	**1.324**	**1.631**	**0.934**

O modelo *Dual-Attention* colaborou para determinar que a complementariedade entre as representações textuais e visuais é significativa para a predição de popularidade. Pois, ao utilizar mecanismos de atenção, essa relação, texto e imagem, é amplificada devido a capturar as melhores representações entre as informações textuais e visuais. Dessa forma, essa abordagem obtém os melhores desempenhos nos três cenários, quando comparado com os resultados obtidos pelo demais modelos presentes na Tabela 13.

6.4 AVALIAÇÃO DO MODELO RAHE COM DIFERENTES REPRESENTAÇÕES DO USUÁRIO

Após compreender a contribuição que cada representação oferece a popularidade e ao analisar que a complementaridade é amplificada utilizando redes de atenção para as modalidades visuais e textuais, foram realizadas análises para as demais abordagens do estado da arte e o modelo RAHE.

A Tabela 14 fornece resultados bastante interessantes, que esclarecem as vantagens em realizar a combinação entre as representações. Foi levado em consideração que os desempenhos dos modelos são similares, caso apresentem uma diferença menor que 2% em relação ao modelo que apresentou melhor desempenho. Tendo os resultados similares destacados em negrito e o melhor desempenho realçado em azul.

Tabela 14 – Comparando os *baselines* com o modelo RAHE com suas diferentes representações do usuário.

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
MLP_{ID}	4.026	1.542	2.733	1.272	2.106	1.104
MLP_{SCL}	2.922	1.335	2.619	1.245	2.876	1.298
MLP_{CTT}	3.056	1.323	2.564	1.220	2.180	1.129
$MLP_{CTT_{sim}}$	2.682	1.277	2.542	1.214	2.180	1.129
LSTM	3.377	1.420	3.393	1.427	1.749	0.976
CNN-VGG	3.963	1.538	3.759	1.489	3.404	1.412
Multi-Modal	3.358	1.440	3.436	1.438	1.657	0.954
Dual-Attention	3.105	1.391	3.051	1.324	1.631	0.934
UHAN	3.474	1.437	2.630	1.256	**1.170**	**0.810**
$RAHE_{SCL}$	2.677	1.277	2.605	1.272	1.504	0.921
$RAHE_{CTT}$	2.804	1.286	**2.499**	**1.219**	**1.189**	**0.817**
$RAHE_{CTT_{sim}}$	**2.478**	**1.218**	**2.483**	**1.215**	**1.187**	**0.817**
$RAHE_{CLS+ID}$	3.120	1.371	2.551	1.235	1.201	0.821
$RAHE_{CTT+ID}$	2.767	1.275	**2.489**	**1.206**	**1.190**	**0.819**
$RAHE_{CTT_{sim}+ID}$	**2.480**	**1.221**	**2.473**	**1.203**	**1.189**	**0.817**
$RAHE_{CLS+CTT}$	2.884	1.317	2.500	1.229	1.228	0.830
$RAHE_{CLS+CTT_{sim}}$	**2.493**	**1.231**	**2.495**	**1.225**	1.228	0.831
$RAHE_{CLS+CTT+ID}$	2.810	1.284	**2.493**	**1.216**	1.248	0.840
$RAHE_{CLS+CTT_{sim}+ID}$	**2.481**	**1.219**	2.501	1.218	1.248	0.839

O modelo *UHAN*, que trata a popularidade para um usuário específico, apresenta um melhor desempenho para o cenário Estratificado. Quando se é comparado *UHAN* e o modelo *Dual-Attention* para o cenário Estratificado, *UHAN* apresenta uma melhoria de 39.4% no MSE e 15.30% no MAE, demostrando a vantagem ao realizar a combinação entre as representações textuais, visuais e do usuário.

Entretanto, apesar de apresentar o melhor resultado para o Estratificado, algo que era esperado, devido prever a popularidade para um usuário específico, esse mesmo modelo não apresentou o mesmo desempenho nos demais cenários, evidenciando assim a limitação desse método, que consegue uma boa diversidade, porém, apresenta pouca capacidade de generalização.

Outro ponto bastante relevante que pode ser notado a partir da Tabela 14, é que todos os modelos que estão destacados, com exceção do modelo *UHAN*, apresentam a abordagem de características contextuais. Os modelos $RAHE_{CTT_{sim}}$ e $RAHE_{CTT_{sim}+ID}$, no contexto geral, apresentaram os melhores resultados, tendo em vista que, ambos apresentaram resultados com uma maior capacidade de generalização e diversidade diante dos

outros modelos.

Mesmo *UHAN* apresentando o melhor desempenho no cenário Estratificado, os modelos $RAHE_{CTT_{sim}}$ e $RAHE_{CTT_{sim+ID}}$ também apresentaram desempenhos aproximados ao seu, onde, $RAHE_{CTT_{sim}}$ apresenta um desempenho inferior a *UHAN* de 1.45% na MSE e 0.86% na MAE, e $RAHE_{CTT_{sim+ID}}$ apresenta um desempenho inferior de 1.62% na MSE e 0.86% na MAE. Demostrando que a abordagem contextual, além de conseguir ter uma maior capacidade de generalização, se assemelha a modelos criados especificamente para lidar com cenários que apresentam diversidade. Em relação ao modelo *UHAN*, o modelo $RAHE_{CTT_{sim}}$ apresentou melhorias de 40.19% no MSE e 17.98% no MAE, para o cenário Disjunto, e 5.92% no MSE e 3.37% no MAE, para o cenário Composto. Já o modelo $RAHE_{CTT_{sim+ID}}$ apresentou melhorias de 40.08% no MSE e 17.69% no MAE, para o cenário Disjunto, e 6.34% no MSE e 4.40% no MAE, para o cenário Composto.

Apesar da combinação entre as representações textuais, visuais e do usuário apresentarem bons resultados, o mesmo não pode ser dito sobre a combinação dos tipos de representações do usuário. Pois, os modelos apresentam bons desempenhos apenas em alguns cenários, não conseguindo balancear a capacidade de generalização e diversidade dos modelos para todos os cenários. Isso pode estar ocorrendo devido a quantidade de informação específica que cada representação do usuário sustenta, podendo atrapalhar os modelos em sua tomada de decisão.

Uma outra análise, se trata do desempenho das características contextuais continuarem a apresentar melhores desempenhos do que a variação modelo *RAHE* que utiliza características sociais. Onde $RAHE_{CTT_{sim}}$ apresenta , em relação $RAHE_{CLS}$, melhorias de 8.03% no MSE e 4.84% no MAE, para o cenário Disjunto, de 4.91% no MSE e 4.69% no MAE, para o cenário Composto, e de 26.70% no MSE e 11.50% no MAE, para o cenário Estratificado. Dessa forma, pode-se observar que mesmo as características contextuais sendo obtidas a partir das características sociais, elas conseguem balancear a diversidade com a capacidade de generalização, enquanto as características sociais apresentam uma grande dificuldade em tratar a popularidade em um cenário com diversidade.

Um dos motivos do seu desempenho se dá ao fato de se ter um contexto comum que engloba usuários com perfis similares, de modo que usuários pertencentes a um mesmo contexto compartilham suas representações visuais e textuais entre si como uma forma de trocar experiências decorridas. Dessa forma, o modelo é capaz de aprender a partir representações compartilhadas pelo contexto, o que geralmente torna um conteúdo popular, ou não, dentro desse contexto específico.

No geral, a combinação entre as representações do usuário não agregou tanto valor e relevância para tratar a popularidade. Mesmo os modelos do $RAHE_{CTT_{sim}}$ e $RAHE_{CTT_{sim+ID}}$ apresentando resultados similares em todos os cenários, o acréscimo do ID, além de aumentar a complexidade do modelo, pois é gerado um *embedding* para o ID com os 341 usuários, não apresenta resultados significantes a ponto do $RAHE_{CTT_{sim+ID}}$ ser escolhido

como modelo mais eficiente. Tendo isso em vista, o modelo $RAHE_{CTT_{sim}}$ se torna mais atrativo para realizar as demais análises, uma vez que, se trata de um modelo mais simples e menos complexo que o $RAHE_{CTT_{sim}+ID}$ e obtêm resultados melhores para os cenários Disjunto e Estratificado, mesmo apresentando uma mínima diferença.

6.5 RELEVÂNCIA DAS REPRESENTAÇÕES TEXTUAIS

Todos os modelos anteriores que utilizaram a representação textual, tiveram como unidade extratora de características a arquitetura do LSTM, utilizando apenas a palavra como fonte de informação textual. Nessa seção serão aplicados o LSTM e a CNN como extrator de características. Além disso, para cada um dos extratores, será disponibilizada a informação textual a nível de palavra e caractere. Dessa forma, são realizadas comparações de desempenhos entre cada uma das abordagens para entender quais benefícios podem ser alcançados para popularidade ao investigar distintas representações textuais.

A Tabela 15 contém os resultados obtidos após as alterações realizadas sobre o modelo *RAHE*. O modelo $RAHE_{CNN_{WORD}+CTT_{sim}}$ apresentou um desempenho superior ao $RAHE_{LSTM_{WORD}+CTT_{sim}}$, que representa o $RAHE_{CTT_{sim}}$, apenas no cenário Composto, tendo um resultado aproximado no cenário Disjunto. Porém, no cenário Estratificado, apresenta um desempenho bastante inferior ao do modelo $RAHE_{LSTM_{WORD}+CTT_{sim}}$, mostrando que ao aplicar a CNN como extrator de características de uma representação textual a nível de palavra, mesmo melhorando o desempenho no cenário Composto, perde sua diversidade e um pouco da sua capacidade de generalização nos demais cenários.

Tabela 15 – Comparando as representações textuais obtidas a partir da CNN e LSTM.

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
$RAHE_{LSTM_{WORD}+CTT_{sim}}$	2.478	1.232	2.483	1.215	1.187	0.820
$RAHE_{CNN_{WORD}+CTT_{sim}}$	2.499	1.225	2.402	1.195	1.288	0.854
$RAHE_{LSTM_{CHAR}+CTT_{sim}}$	**2.409**	**1.214**	**2.353**	**1.169**	1.239	0.836
$RAHE_{CNN_{CHAR+}+CTT_{sim}}$	2.453	1.226	2.374	1.188	**1.185**	**0.819**

Entretanto, quando se é analisado os resultados obtidos pelos modelos que utilizam os caracteres como fonte de informação textual para o modelo, os resultados são superiores aos que foram obtidos para os modelos que utilizam a palavra como fonte de informação textual em quase todos os cenários. O $RAHE_{LSTM_{CHAR}+CTT_{sim}}$ apresenta desempenhos superiores ao $RAHE_{LSTM_{WORD}+CTT_{sim}}$ com melhorias de 2.86% no MSE e 1.48% no MAE, para o cenário Disjunto, e de 5.52% no MSE e 3.93% no MAE, para o cenário Composto. No entanto, apresenta um resultado bastante inferior ao modelo $RAHE_{LSTM_{CHAR}+CTT_{sim}}$ no cenário Estratificado, perdendo na diversidade do modelo.

Por outro lado, apesar do modelo $RAHE_{LSTM_{CHAR}+CTT_{sim}}$ apresentar melhores desempenho para os cenários Disjunto e Composto, o $RAHE_{CNN_{CHAR}+CTT_{sim}}$ apresenta resultados mais estáveis em todos os cenários, apresentando desempenhos superiores em todos os cenário em relação ao modelo $RAHE_{LSTM_{WORD}+CTT_{sim}}$. O $RAHE_{CNN_{CHAR}+CTT_{sim}}$ apresenta desempenhos superiores ao $RAHE_{LSTM_{WORD}+CTT_{sim}}$ com melhorias de 2.86% no MSE e 0.90% no MAE, no cenário Disjunto, e de 4.59% no MSE e 3.93% no MAE, para o cenário Composto. Porém, para o cenário Estratificado apresentou uma melhoria mínima de 0.16% no MSE e 0.12% no MAE.

A partir desses resultados é possível notar que o uso de caracteres como informação textual auxiliou bastante no desempenho do modelo, melhorando os resultados para os cenários Disjunto, Estratificado e, principalmente, o Composto. As incorporações de palavras no nível do caractere podem ser particularmente significativas no contexto de predição de popularidade, uma vez que novos nomes de entidades são criados com frequência e podem seguir padrões consistentes, como prefixos ou sufixos comuns que são relevantes para o problema de popularidade, ou até mesmo *emoticons*.

Um outro benefício de utilizar caracteres é que o modelo pode aprender erros de ortografia presentes na representação textual, uma vez que se está lidando com um problema de mídias sociais, esse tipo de situação é bastante corriqueiro, pois, os usuários não se importam tanto com erros ortográficos, podendo gerar palavras desconhecidas que não são entendidas pelo modelo. Ao usar uma representação a nível de caractere é possível lidar com esse problema naturalmente, pois cada palavra é considerada como não mais do que uma composição de letras individuais. Dessa forma, o problema enfrentado pela representação textual a nível de palavras, que necessitam de grandes vocabulários, de aparecer uma palavra fora do vocabulário, também conhecido como *out-of-vocabulary* (OOV), é evitado com o uso de representação textual a nível de palavra.

Em relação ao uso da LSTM ou CNN como extrator de características, depende muito do problema que está sendo lidado. As LSTMs, podem capturar dependências entre palavras, mesmo que estejam distantes umas das outras, entretanto, para a popularidade, a relação entre as palavras pode não influenciar tanto quanto a existência de uma palavra. CNNs também são eficientes em termos de representação, filtros convolucionais aprendem boas representações automaticamente, sem precisar representar todo o vocabulário, capturando recursos bastante similares a n-gramas, mas os representam de uma maneira mais compacta. Um grande argumento para as CNNs é que elas, além de tudo, são rápidas e reduzem a complexidade de treinamento, diferentemente da LSTM, como demostrado por Zhai, Nguyen e Verspoor (2018).

6.6 AVALIAÇÃO DO MÉTODO DE CORREÇÃO DE ERRO – RAHE-CE

Nessa etapa é realizada uma avaliação da contribuição que o método de correção de erro pode trazer a um modelo. O objetivo desse método é aprender, minimizar e ajustar o erro

obtido pelo modelo *RAHE*, principalmente, os erros relacionados as amostras de baixa frequência no conjunto amostral. Os resultados após a integração do modelo *RAHE* com o método de correção de erro são apresentados na Tabela 16. A partir desses resultados, é possível notar a influência que esse módulo causou nos desempenhos dos modelos. Todos os modelos que apresentaram esse módulo integrado tiveram resultados com desempenhos bastante superiores em todos os cenários.

Tabela 16 – Desempenho dos modelos após a integração do método de correção de erro.

	Disjunto		Composto		Estratificado	
	MSE	MAE	MSE	MAE	MSE	MAE
$RAHE_{LSTM_{WORD}+CTT_{sim}}$	2.478	1.232	2.483	1.215	1.187	0.820
$RAHE_{CNN_{WORD}+CTT_{sim}}$	2.499	1.225	2.402	1.195	1.288	0.854
$RAHE_{LSTM_{CHAR}+CTT_{sim}}$	2.409	1.214	2.353	1.169	1.239	0.836
$RAHE_{CNN_{CHAR+}+CTT_{sim}}$	2.453	1.226	2.374	1.188	1.185	0.819
$RAHE\text{-}CE_{LSTM_{WORD}+CTT_{sim}}$	2.413	1.213	2.395	1.193	1.125	0.792
$RAHE\text{-}CE_{CNN_{WORD}+CTT_{sim}}$	2.422	1.208	2.345	1.178	1.167	0.808
$RAHE\text{-}CE_{LSTM_{CHAR}+CTT_{sim}}$	2.369	1.199	**2.210**	**1.132**	1.120	0.788
$RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$	**2.345**	**1.193**	2.213	1.150	**1.082**	**0.777**

Apesar dos modelos $RAHE\text{-}CE_{LSTM_{WORD}+CTT_{sim}}$ e $RAHE\text{-}CE_{CNN_{WORD}+CTT_{sim}}$ terem melhorado seus desempenhos em relação aos seus modelos sem a correção de erro, os modelos $RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$ e $RAHE\text{-}CE_{LSTM_{CHAR}+CTT_{sim}}$ apresentaram os maiores desempenhos no contexto geral. Porém, o modelo $RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$ apresenta melhores desempenhos que o $RAHE\text{-}CE_{LSTM_{CHAR}+CTT_{sim}}$ nos cenários Disjunto e Estratificado, tendo o Composto resultados similares, tornando assim o modelo $RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$ com o melhor desempenho.

Em relação ao modelo $RAHE_{CNN_{CHAR+}+CTT_{sim}}$, o modelo $RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$ apresentou melhorias de 4.60% no MSE e 2.76% no MAE, para o cenário Disjunto, de 7.27% no MSE e 4.94% no MAE, para o cenário Composto, e de 9.51% no MSE e 5.40% no MAE, para o cenário Estratificado. Dessa forma, o modelo $RAHE\text{-}CE_{CNN_{CHAR}+CTT_{sim}}$ se trata da versão final do modelo $RAHE$ que é composto pelas características contextuais, representação textual a nível de caractere, com a CNN como unidade extratora de características, e o método de correção de erro.

6.7 VISUALIZAÇÃO DO EMBEDDING CONTEXTUAL

Em busca de entender melhor as características contextuais obtidas, foi analisada a relação que existe entre o *embedding* contextual, aprendido durante o treinamento do modelo de popularidade, e a popularidade dos usuários pertencentes a um determinado contexto.

Para isso, foi obtido o valor médio da popularidade de cada *embedding*, presente no *embedding* contextual, e a partir desses valores médios, cada *embedding* foi associado a um nível de popularidade. Dessa forma, alguma tendência existente entre os níveis de popularidade e o *embedding* contextual pode ser observada.

O nível de popularidade foi definido a partir da análise feita sobre o histograma da popularidade das publicações em conjunto com a função de distribuição gaussiana, como demostrado na Figura 32. A partir do histograma, é possível visualizar quais são valores mais comuns de popularidade concentrados em uma determinada região, e assim, definir níveis de popularidade para valores que estejam contidos dentro dessas regiões. Então, foram obtidos cinco níveis de popularidade que representam bem a distribuição da popularidade.

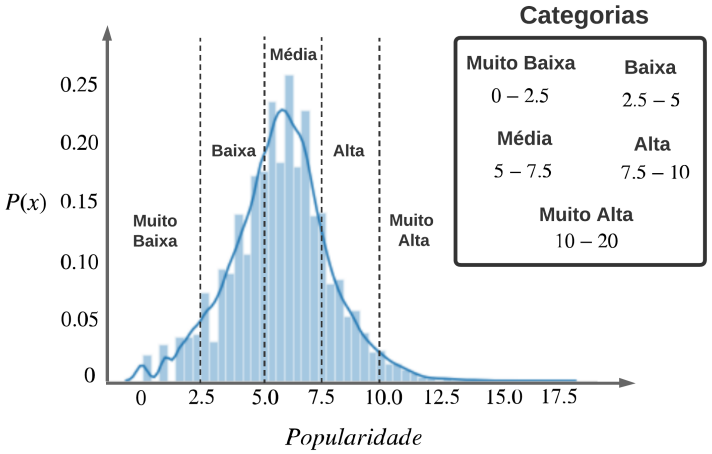


Figura 32 – Divisão dos valores de popularidade em cinco níveis: Muito baixa, baixa, média , alta e muito alta.

Com o auxílio da Incorporação de Vizinhos Estocásticos Distribuídos (*t-Distributed Stochastic Neighbor Embedding* - t-SNE), que é uma técnica para redução de dimensionalidade que é particularmente adequada para a visualização de conjuntos de dados de alta dimensão (MAATEN; HINTON, 2008), cada *embedding* teve sua dimensionalidade reduzida de 512 para duas dimensões, facilitando assim a visualização de cada *embedding* no espaço contextual.

Dessa forma, cada *embedding* contextual obtido para os cenários Disjunto, Composto e Estratificado foi plotado e seus gráficos são apresentados nas Figuras 33, 34 e 35, respectivamente. A partir dos *embeddings* contextuais gerados para cada um dos cenários, é possível visualizar uma clusterização dos dados baseado nos valores médios das popularidades de cada contexto.

A princípio, pode-se perceber que a partir de características sócias do perfil, *avgview*, *menbercount* e *groupcount*, é possível saber antecipadamente, a partir do contexto em que esse usuário será inserido, qual a popularidade média de um novo conteúdo, uma

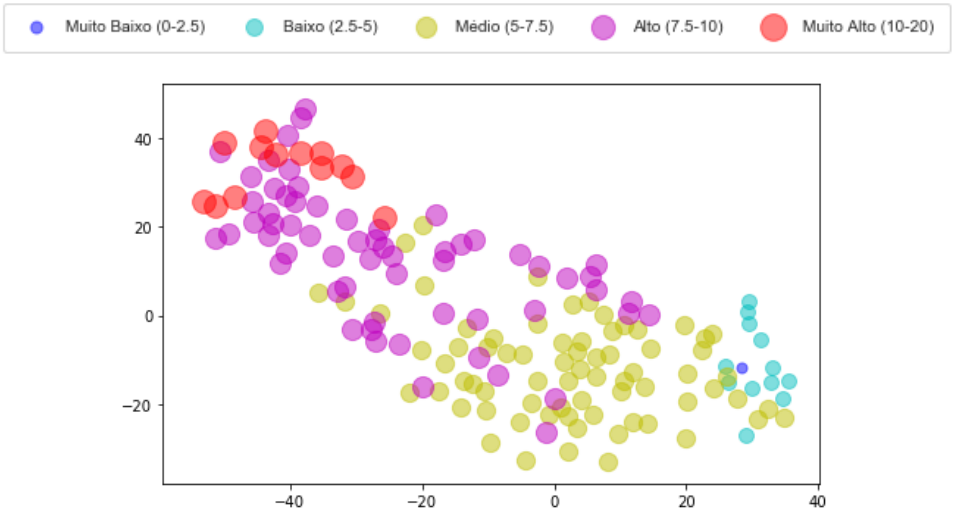


Figura 33 – Visualização do *embedding* contextual para o cenário Disjunto a partir do t-SNE.

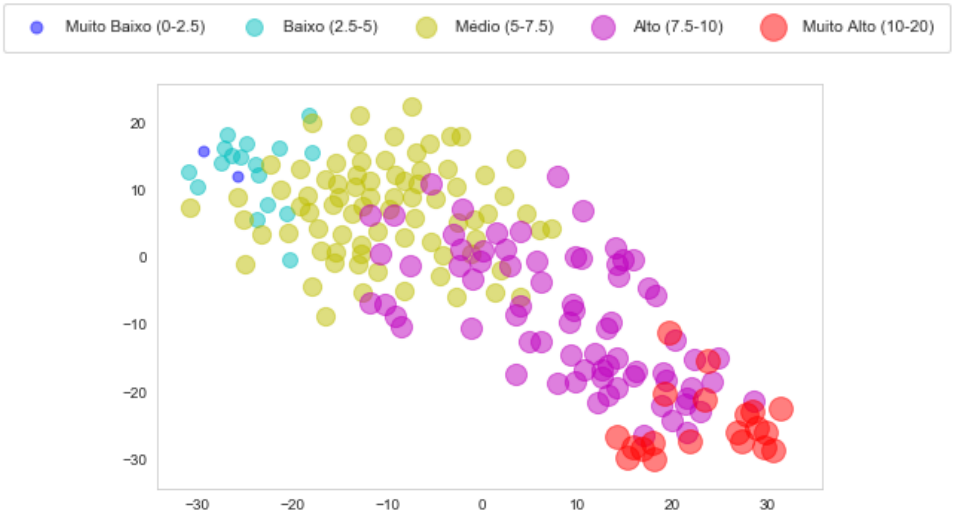


Figura 34 – Visualização do *embedding* contextual para o cenário Composto a partir do t-SNE.

vez que, o contexto é gerado a partir dessas características que estão presentes em todos os perfis. Apesar de outras informações também influenciarem no processo de predição de popularidade, como a informação visual e textual que são compartilhadas por cada contexto, entretanto, o *embedding* contextual nos dá uma noção de uma média de popularidade que pode ser obtida para um novo conteúdo publicado por um usuário presente em um determinado contexto.

Além disso, a partir da visualização do *embedding* contextual é possível comprovar o porquê do êxito obtido pela abordagem que utiliza a similaridade entre contextos. Pois, os clusters concentram contextos com valores de média de popularidade similares próximos uns aos outros no espaço contextual. Nesse caso, o valor médio se trata da informação comum existente entre a proximidade dos contextos no espaço, mostrando a correlação

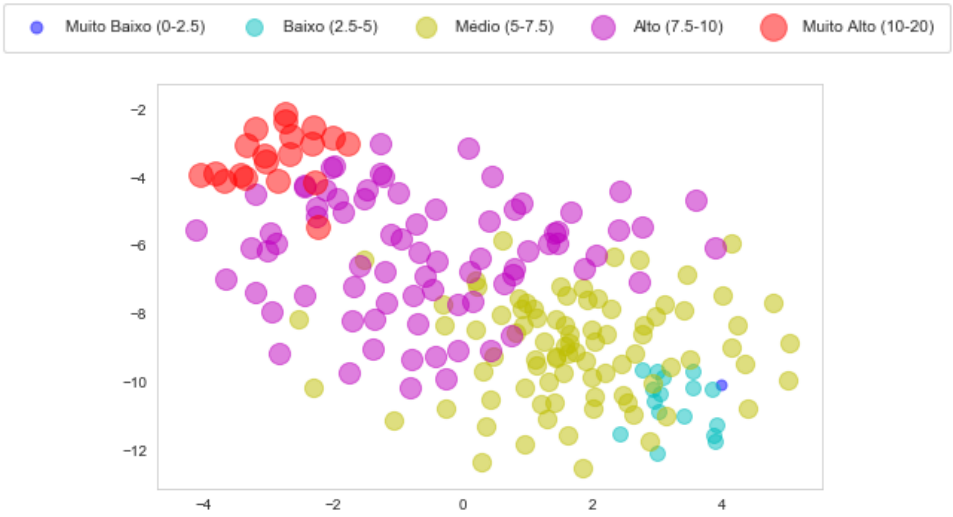


Figura 35 – Visualização do *embedding* contextual para o cenário Estratificado a partir do t-SNE.

que existe entre os contextos e o valor médio da popularidade. Dessa forma, caso um usuário não apresente um contexto específico para ele, é possível associá-lo a um contexto mais próximo do que seria obtido por ele, visto que, os valores de popularidade média entre esses contextos seriam bem próximos no espaço contextual.

6.8 VISUALIZAÇÃO DO COMPORTAMENTO DA CORREÇÃO DE ERRO

Para finalizar, foram realizadas análises sobre o efeito que a aplicação do modelo de correção de erro ocasiona no resultado final. Como levantado na Seção 4.6, a probabilidade de ocorrer valores nos extremos da curva é muito baixa. Geralmente, em problemas de regressão a maior dificuldade apresentada pelos modelos é prever esses valores mais extremos, valores que não aparecem com uma certa frequência. Pois, o modelo terá uma certa dificuldade em aprender suas características a modo de lhes dar um resultado mais adequado. Devido a isso, a tendência é que os erros nos extremos sejam maiores, como mostrado na Figura 36.

Dessa forma foi analisado para cada cenário o comportamento da curva de erro médio obtida a partir da correção de erro, e também foi utilizado o diagrama de caixa, que avalia a variação dos dados, para fornecer uma análise visual da mediana, dispersão, simetria e valores discrepantes (*outliers*) do conjunto de dados. As medidas de estatística descritiva como o mínimo, máximo, primeiro quartil, segundo quartil ou mediana e o terceiro quartil formam o diagrama de caixa, como demostrado na Figura 37.

Essa análise irá focar mais nas medidas de dispersão dos dados, que pode ser representada pelo intervalo interquartílico e pela amplitude, e na mediana ou medida de tendência central, pois, é mais indicada quando os dados possuem distribuição assimétrica, uma vez que a média aritmética é influenciada por valores extremos. O intervalo interquartílico é

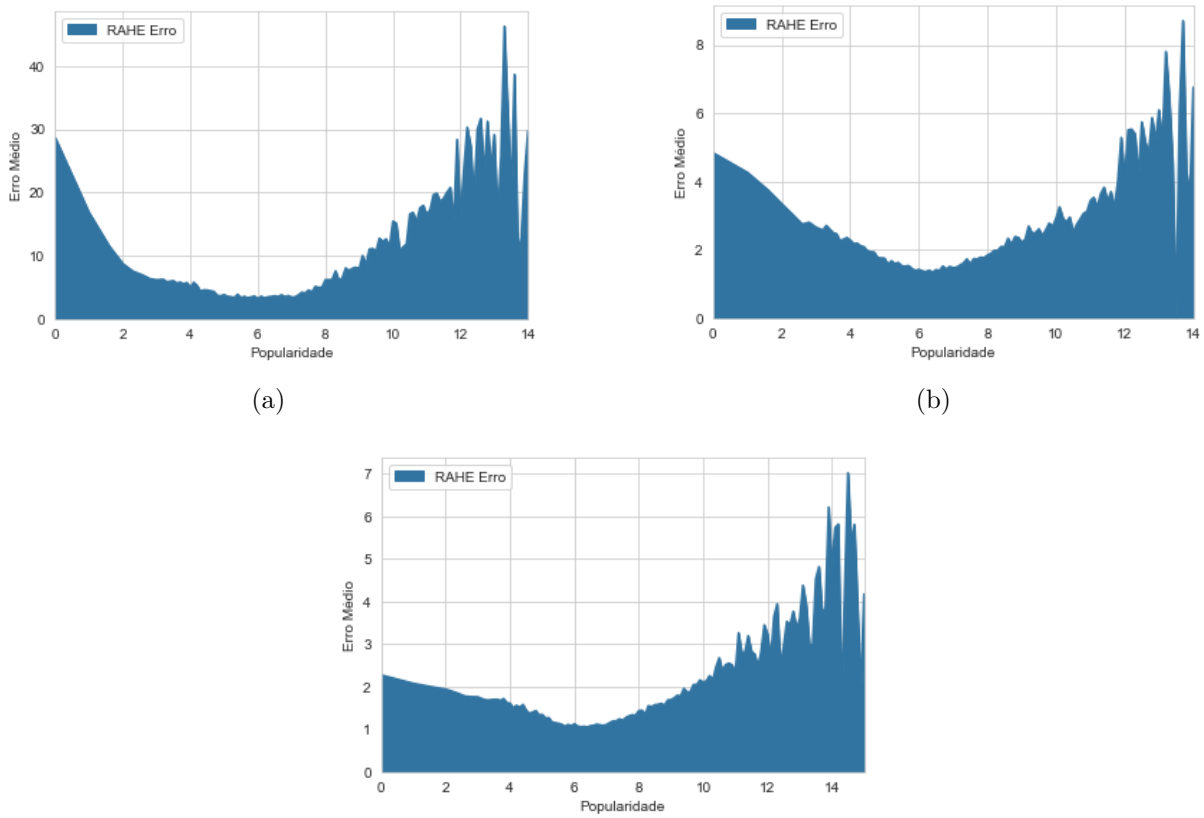


Figura 36 – Erros médios da popularidade: (a) Erro médio no cenário Disjunto, (b) Erro médio no cenário Composto e (c) Erro médio no cenário Estratificado

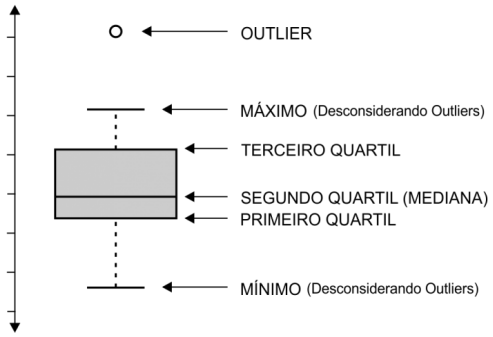


Figura 37 – Medidas descritivas que compõe o diagrama de caixa.

a diferença entre o terceiro quartil e o primeiro quartil (tamanho da caixa), já amplitude é obtida a partir da diferença entre valor máximo e valor mínimo. Embora a amplitude seja de fácil entendimento, o intervalo interquartílico é uma estatística mais robusta para medir variabilidade, uma vez que não sofre influência de *outliers*.

6.8.1 Erro médio do cenário Disjunto

A partir da Figura 38 é possível observar que o método de correção de erro apresentou uma redução na média de erro apresentada pelo modelo *RAHE*. Entretanto, essa redução não aconteceu de maneira uniforme, tendo os valores médios de erros presentes no extremo direito da curva um pequeno aumento na média do erro, não tão perceptível quanto a diminuição do erro médio para os valores no extremo esquerdo da curva.

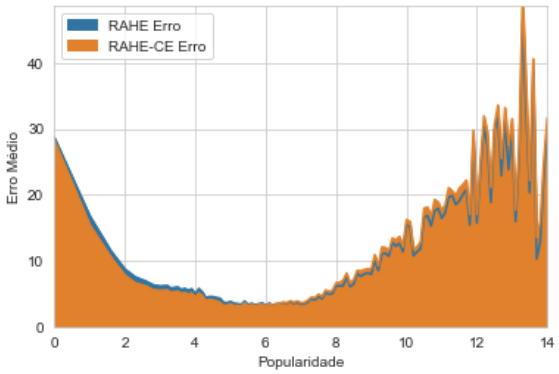


Figura 38 – Comparação entre o erro médio obtido pelo modelo *RAHE versus RAHE-CE* no cenário Disjunto

Para se ter uma informação mais detalhada dos resultados, foi analisado o diagrama de caixa para alguns intervalos de valores de popularidade presentes nos cinco níveis de popularidade definidos na Seção 6.7, para analisar a dispersão e a medida de tendência central da distribuição dos valores. Na Figura 39 pode-se observar que o método de correção apresentou, tanto para amplitude como para o intervalo interquartílico, reduções consideráveis no diagrama que representa o valores da categoria "muito baixo", assim como a medida de tendência central.

No diagrama que representa a categoria "baixa", também apresenta reduções, porém, a medida de tendência central já não sofre tanta alteração. Nos demais diagramas a tendência central permanece contínua e começa a apresentar pequenas variações, não só na tendência, como nos valores de amplitude e intervalo interquartílico. Demostrando assim, que o método de correção lida bem com os valores de popularidade mais baixos, situado no extremo esquerdo da curva da Figura 38, porém, apresenta ainda uma certa deficiência em ajustar o erro para valores mais elevados de popularidade.

6.8.2 Erro médio do cenário Composto

Para o cenário composto é possível notar que houve uma variação maior na correção do erro médio realizada pelo modelo *RAHE-CE*, para os valores do extremo esquerdo da curva da Figura 40. No entanto, houve um pequeno acréscimo na média do erro do modelo *RAHE-CE* no extremo direito da curva quando comparado com o modelo *RAHE*, uma

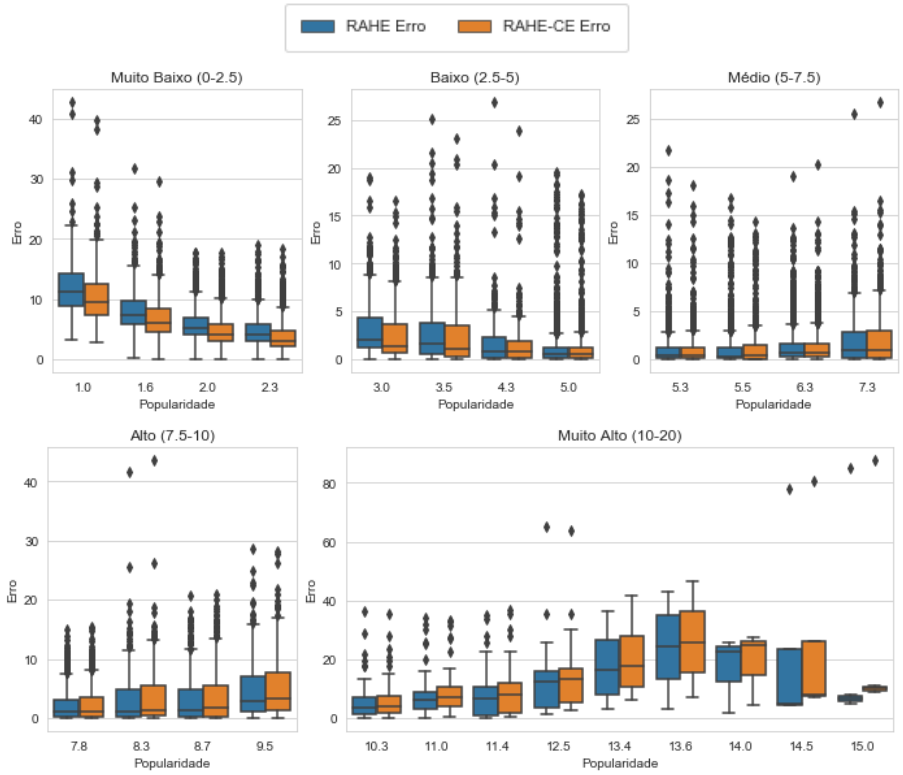


Figura 39 – Diagrama de caixa sobre intervalos de valores de popularidade para o cenário Disjunto.

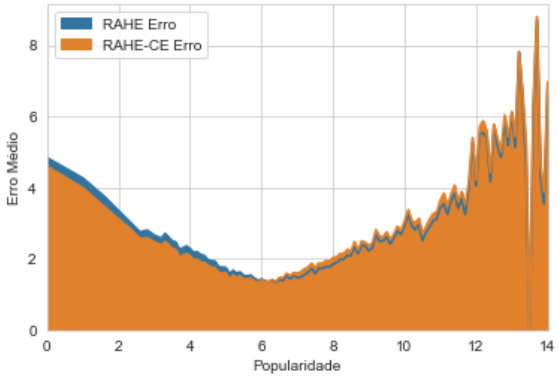


Figura 40 – Comparação entre o erro médio obtido pelo modelo *RAHE versus RAHE-CE* no cenário Composto

situação bastante parecida com a obtida pelo cenário Disjunto, porém, com flutuações mais aparentes para valores de popularidade mais elevadas.

A partir do diagrama de caixa da Figura 41, é possível notar que o método de correção apresentou variações consideráveis, tanto para amplitude como para o intervalo interquartílico, para valores de popularidade mais baixos. Porém, a medida de tendência central não sofreu alterações consideráveis nos primeiros diagramas. Entretanto, nos diagramas para as categorias "alta"e "muito alta", não só a tendência central, como os valores de amplitude e intervalo interquartílico, sofreram variações mais bruscas, do que as apresentadas pelo

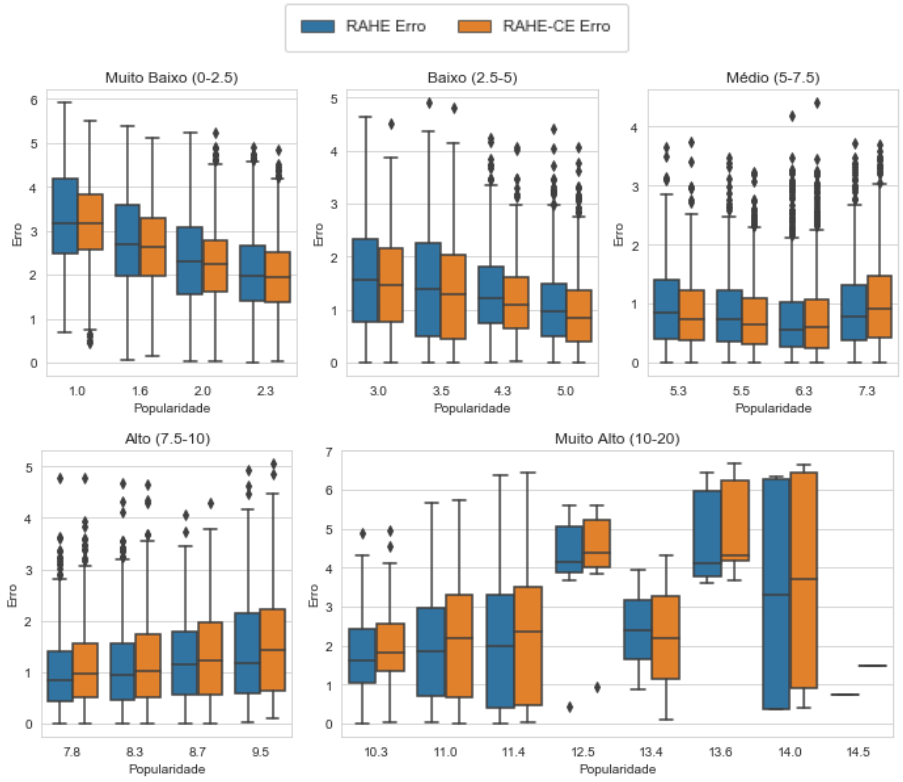


Figura 41 – Diagrama de caixa sobre intervalos de valores de popularidade para o cenário Composto.

modelo *RAHE-CE* no cenário Disjunto.

6.8.3 Erro médio do cenário Estratificado

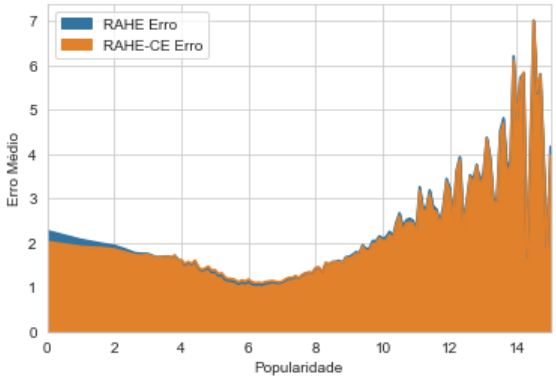


Figura 42 – Comparação entre o erro médio obtido pelo modelo *RAHE versus RAHE-CE* no cenário Estratificado

O cenário Estratificado foi o que apresentou um comportamento mais uniforme, tendo o modelo *RAHE-CE* uma maior variação, não só no erro médio obtido para valores no extremo esquerdo da curva da Figura 42, mas também, para os valores médios obtidos para os valores no extremo direto da curva.

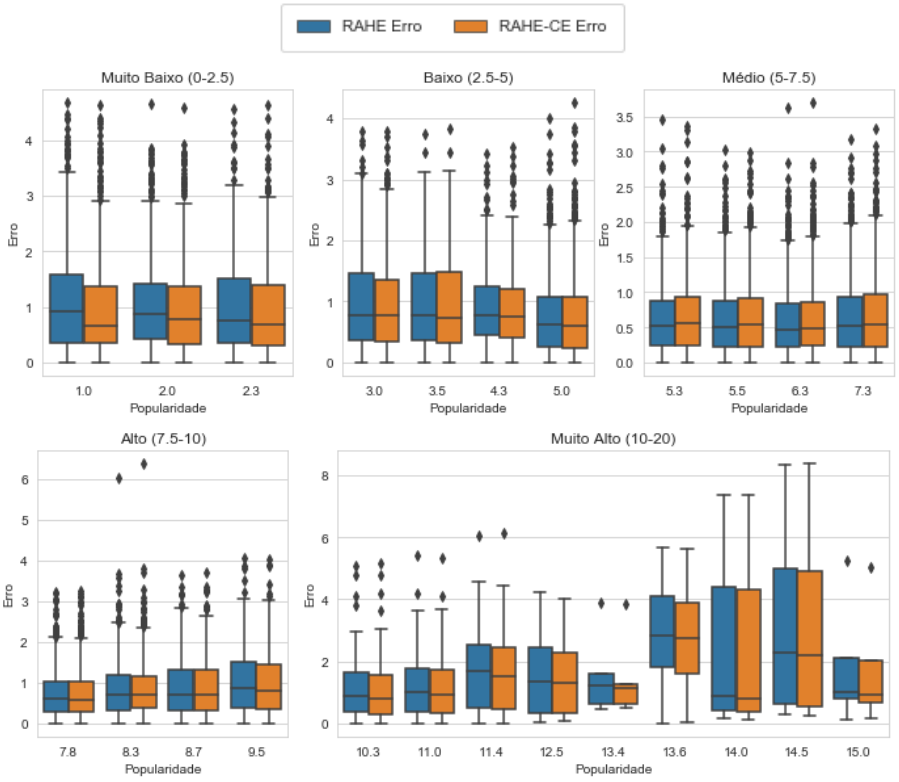


Figura 43 – Diagrama de caixa sobre intervalos de valores de popularidade para o cenário Estratificado.

Essa análise feita sobre a curva obtida pelo modelo *RAHE-CE*, fica ainda mais nítida quando se é analisado com o diagrama de caixa da Figura 43. Onde é possível notar que o modelo *RAHE-CE* apresentou variações positivas na amplitude, intervalo interquartílico e na medida de tendência central para as popularidade nos extremos esquerdo e direito da curva. Porém, não apresentou o mesmo desempenho nos valores médios de popularidade, apresentando algumas variações negativas em relação ao modelo *RAHE*, mas, variações com flutuações mais discretas, pouco significativas.

7 CONCLUSÕES

A variação $RAHE_{CTT_{sim}}$ conseguiu explorar perfeitamente os aspectos de capacidade de generalização e diversidade, obtendo resultados bastante expressivos em relação aos modelos presentes no estado da arte e demostrando que a aplicação de um contexto comum entre os usuários é bastante promissora. Ao analisar a contribuição individual das representações, percebeu-se que, a proposta de um contexto compartilhado entre os usuários apresenta alta capacidade de generalização assim como de diversidade, apresentando resultados equiparáveis a modelos construídos apenas para atacar um desses aspectos.

Apesar da abordagem contextual ser obtida a partir da manipulação de características sociais do perfil, apresentam resultados satisfatórios em cenários que lidam com a capacidade de generalização. O contexto, além de melhorar a capacidade de generalização, apresentou uma ótima capacidade de lidar com a diversidade dos dados, algo que as características sociais que o constituem não conseguem lidar, aumentando o desempenho obtido pelas características sócias em 26.70% no mse e 11.50% no mae, para o cenário Estratificado.

O modelo $RAHE_{CTT_{sim}}$ superou o modelo base UHAN, nos cenários Disjunto e Composto. Ao apresentar um resultado semelhante ao UHAN no cenário Estratificado, o modelo $RAHE_{CTT_{sim}}$ demostra que se equipara a abordagens que foram definidas para atacar especificamente a diversidade, e que é possível obter um modelo que consegue se adequar perfeitamente aos dois aspectos.

O modelo UHAN cria um *embedding* dos usuários que utiliza os 341 usuários presentes na base, enquanto o $RAHE_{CTT_{sim}}$ cria um *embedding* contextual com 200 contextos, no qual, esses 341 usuários estão imersos. Dessa forma, se aparecer um novo usuário, o modelo de UHAN não consegue mapeá-lo no *embedding* criado, diferentemente da abordagem contextual que mapeia o usuário para um contexto comum pertencente ao *embedding* contextual.

A partir das contribuições individuais também foi possível perceber a importância das informações textuais para tratar o problema de diversidade, apresentando resultados expressivos em relação a modelos que utilizam dados que são específicos de cada usuário. Mostrando que outros aspectos influenciam na popularidade, não só apenas o usuário em si, mas o tipo de conteúdo que está sendo apresentado, e que investigar características textuais específicas para o problema de popularidade é de suma importância.

Dessa forma, foi observado que os desempenhos obtidos pelos modelos que utilizam representações textuais a nível de caracteres são superiores aos obtidos pelas representações textuais a nível de palavras. Além de tudo, conseguem manter a capacidade de generalização e diversidade do modelo $RAHE_{CTT_{sim}}$. As incorporações de palavras no nível do caractere podem ser particularmente significativas no contexto de predição de

popularidade, uma vez que novos nomes de entidades são criados com frequência e podem seguir padrões consistentes, como prefixos ou sufixos comuns que são relevantes para o problema de popularidade.

Em relação ao uso da LSTM ou CNN como extratores de características, foi visto que a CNN apresentou melhores desempenho que a LSTM ao extrair características a nível de caracteres. Entretanto, isso depende muito do problema em que estamos lidando, pois, as LSTMs podem capturar dependências entre palavras, mesmo que estejam distantes umas das outras, entretanto, para a popularidade a relação entre as palavras ou caracteres pode não influenciar tanto quanto a frequência de alguns padrões consistentes. CNNs são eficientes em termos de representação de padrões, filtros convolucionais aprendem boas representações automaticamente, sem precisar representar todo o vocabulário, capturando recursos bastante similares a *n-grams*, mas os representam de uma maneira mais compacta. Um grande argumento para as CNNs é que elas, além de tudo, são rápidas e reduzem a complexidade de treinamento, diferentemente da LSTM.

Ao adicionar o método de correção de erro, gerando a variação RAHE-CE, foram obtidos desempenhos superiores aos que já haviam sido obtidos. A ideia por trás dessa abordagem é criar um modelo que aprenda os erros cometidos pelo modelo $RAHE_{CTT_{sim}}$ e os corrija a modo de suavizar o erro médio do modelo, principalmente, os erros relacionados as amostras de baixa frequência no conjunto amostral. E a partir de uma modelo simples de duas camadas de redes *feedfoward*, alimentado com as informações da representação do usuário e com a saída do modelo RAHE, foi possível obter um modelo que conseguiu minimizar o erro médio da saída do modelo RAHE.

Dessa forma, a versão final do modelo RAHE, composto pelas características contextuais, representação textual a nível de caractere com a CNN como unidade extratora de características e um método de correção de erro, superou o modelo base UHAN em 48.14% no *mse* e 20.45% no *mae*, para o cenário Disjunto, em 18.84% no *mse* e 9.21% no *mae*, para o cenário Composto (que foi introduzido pela abordagem *UHAN* (ZHANG et al., 2018)), e de 8.13% no *mse* e 4.24% no *mae*, para o cenário Estratificado. Demostrando que além de ter uma excelente capacidade de generalização, o modelo consegue superar o desempenho de diversidade apresentado pelas outras abordagens específicas para esse tipo de aspecto.

Além disso tudo, a partir do *embedding contextual*, gerado para cada um dos cenários, foi visualizado que os contextos ficaram agrupados em clusters bem característicos separados por nível de popularidade. Constando a relação existente entre o contexto comum, compartilhados pelos usuários, e o nível de popularidade, demostrando o porquê do êxito obtido pela abordagem que utiliza a similaridade entre contextos. Pois, usuários com contextos próximos, apresentam valores de popularidade média, entre esses contextos, bem próximos no espaço contextual. Foi visto também, que a partir de informações de características sociais, como *avgview*, *menbercount* e *groupcount*, pode-se saber ante-

cipadamente, a partir do contexto em que esse usuário será inserido, qual a popularidade média que um novo conteúdo irá apresentar, uma vez que, o contexto é gerado a partir dessas informações primárias.

Apesar dos bons resultados obtidos pelo método de correção, ainda existe margem para melhorias. Pois, ao analisar os gráficos da média de erro para cada um dos cenários, após sua aplicação, foi possível notar que há uma melhoria do erro médio para valores mais baixos e médios de popularidade, enquanto, para valores muito altos, há um pequeno acréscimo do erro médio, mostrando uma certa fragilidade desse tipo de abordagem ao tratar com valores de popularidade mais alta. Entretanto, mesmo ocorrendo essas variações para valores de popularidade mais elevados, essas variações não foram significantes, pois não interferiram no desempenho final obtido pelo módulo de revisão de erro. Mas vale ressaltar, que é necessário procurar entender o porquê desse comportamento para valores mais elevados de popularidade.

No geral, a versão final do modelo RAHE apresentou resultados que superam modelos no estado da arte na predição de popularidade, em todos os cenários e aspectos que podem ser abordados para o problema tratado, a partir de um contexto comum entre os usuários, representação textuais a nível de caractere, tendo CNN como extrator de características, e um método de correção de erro. Demostrando que o estudo realizado é relevante para o problema de predição de popularidade, alcançando um desempenho significativo de 48.14% no mse no cenário Disjunto, cenário que reflete bem um ambiente real.

Tendo como as principais contribuições dessa dissertação: a concepção de um contexto compartilhado entre os usuários de características sociais em comum; um método de correção de erro que tem como propósito minimizar os erros obtidos pela predição de popularidade; a análise da contribuição de informações textuais a nível de caractere; a correlação existente entre a popularidade média e o contexto comum; e a definição de cenários que garantem a cobertura da diversidade e capacidade de generalização dos modelos

7.1 TRABALHOS FUTUROS

Existem algumas melhorias que podem ser aplicadas sobre o modelo RAHE obtido:

- Realizar de maneira automatizada a discretização de variáveis contínuas para obtenção de contexto comum entre os usuários. Pois, o k utilizado para obter os grupos de cada variável contínua é obtido empiricamente a partir da análise visual do histograma obtido pela discretização.

- Analisar o uso de outras métricas de similaridade para obter o contexto mais similar a um determinado usuário. Pois, foi utilizada a distância euclidiana, uma medida mais simples, apenas para provar que a abordagem de similaridade contextual adotada é coerente.

- Melhorar a performance do método de correção de erro, visando a minimização do erro obtido para valores mais altos de popularidade e o entendimento do seu comportamento para valores mais altos de popularidade.
- Estender o modelo RAHE para outras mídias sociais. Como essa abordagem precisa apenas de características sociais do perfil, é possível aplicá-la a outras mídias sociais utilizando apenas as informações sociais dessas outras mídias, não se limitando apenas a uma única plataforma.
- Estender o modelo RAHE para lidar com conteúdos que apresentem informação textual em português. .

REFERÊNCIAS

ALOM, M. Z.; TAHA, T. M.; YAKOPCIC, C.; WESTBERG, S.; HASAN, M.; ESESN, B. C. V.; AWWAL, A. A. S.; ASARI, V. K. The history began from alexnet: A comprehensive survey on deep learning approaches. *CoRR*, abs/1803.01164, 2018.

ALOUFI, S.; ZHU, S.; El Saddik, A. On the prediction of flickr image popularity by analyzing heterogeneous social sensory data. *Sensors (Switzerland)*, v. 17, n. 3, p. 1–27, 2017. ISSN 14248220.

ANTOL, S.; AGRAWAL, A.; LU, J.; MITCHELL, M.; BATRA, D.; ZITNICK, C. L.; PARIKH, D. VQA: visual question answering. *CoRR*, abs/1505.00468, 2015.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. *ArXiv*, v. 1409, 09 2014.

BENGIO, Y.; DUCHARME, R.; VINCENT, P.; JANVIN, C. A neural probabilistic language model. *J. Mach. Learn. Res.*, JMLR.org, v. 3, p. 1137–1155, mar. 2003. ISSN 1532-4435.

BENGIO, Y.; SIMARD, P.; FRASCONI, P. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 5, n. 2, p. 157–166, mar. 1994. ISSN 1045-9227.

BERZAL, F.; CUBERO, J.-C.; MARíN, N.; SáNCHEZ, D. Building multi-way decision trees with numerical attributes. *Inf. Sci.*, v. 165, p. 73–90, 09 2004.

BORTH, D.; JI, R.; CHEN, T.; BREUEL, T.; CHANG, S.-F. Large-scale visual sentiment ontology and detectors using adjective noun pairs. In: *Proceedings of the 21st ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2013. (MM '13), p. 223–232. ISBN 978-1-4503-2404-5.

BOUREAU, Y.; BACH, F.; LECUN, Y.; PONCE, J. Learning mid-level features for recognition. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2010*. [S.l.: s.n.], 2010. p. 2559–2566. ISBN 9781424469840.

BOUVRIE, J. Notes on convolutional neural networks. 2006.

BRÉBISSON, A. de; SIMON, É.; AUVOLAT, A.; VINCENT, P.; BENGIO, Y. Artificial neural networks applied to taxi destination prediction. *CoRR*, abs/1508.00021, 2015.

BRITZ, D.; GOLDIE, A.; LUONG, M.; LE, Q. V. Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906, 2017.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2016. (KDD '16), p. 785–794. ISBN 978-1-4503-4232-2.

CHLEBUS, B. S.; NGUYEN, S. H. On finding optimal discretizations for two attributes. In: *Proceedings of the First International Conference on Rough Sets and Current Trends in Computing*. London, UK, UK: Springer-Verlag, 1998. (RSCTC '98), p. 537–544. ISBN 3-540-64655-8.

COLLOBERT, R.; WESTON, J.; BOTTOU, L.; KARLEN, M.; KAVUKCUOGLU, K.; KUKSA, P. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, JMLR.org, v. 12, p. 2493–2537, nov. 2011. ISSN 1532-4435.

ELMAN, J. L. Finding structure in time. *Cognitive Science*, v. 14, n. 2, p. 179 – 211, 1990. ISSN 0364-0213.

FAN, C. Survey of convolutional neural network. In: . [S.l.: s.n.], 2016.

FAYYAD, U. M.; IRANI, K. B. Multi-interval discretization of continuous-valued attributes for classification learning. In: *IJCAI*. [S.l.: s.n.], 1993. p. 1022–1029.

GARCIA, S.; LUENGO, J.; SAEZ, J. A.; LOPEZ, V.; HERRERA, F. A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning. *IEEE Trans. on Knowl. and Data Eng.*, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 25, n. 4, p. 734–750, abr. 2013. ISSN 1041-4347.

GELLI, F.; URICCHIO, T.; BERTINI, M.; BIMBO, A. D.; CHANG, S.-F. Image popularity prediction in social media using sentiment and context features. In: *Proceedings of the 23rd ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2015. (MM '15), p. 907–910. ISBN 978-1-4503-3459-4.

GERS, F. A.; SCHMIDHUBER, J. Recurrent nets that time and count. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. [S.l.]: IEEE, 2000. v. 3, p. 189–194 vol.3.

GUO, C.; BERKHAHN, F. Entity embeddings of categorical variables. *CoRR*, abs/1604.06737, 2016.

HARIDAS, R.; JYOTHI, R. L. Convolutional neural networks: A comprehensive survey. *International Journal of Applied Engineering Research*, v. 14, n. 3, p. 780–789, 2019.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.

HEMADA, B.; LAKSHMI, K. A Study On Discretization Techniques. v. 2, n. 8, p. 1887–1892, 2013.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 9, n. 8, p. 1735–1780, nov. 1997. ISSN 0899-7667.

HUANG, G.; LIU, Z.; WEINBERGER, K. Q. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

JORDAN, M. I. *Serial Order: A Parallel, Distributed Processing Approach*. [S.l.], 1986.

KALCHBRENNER, N.; GREFENSTETTE, E.; BLUNSOM, P. A convolutional neural network for modelling sentences. *CoRR*, abs/1404.2188, 2014.

KARPATHY, A.; LI, F. Deep visual-semantic alignments for generating image descriptions. *CoRR*, abs/1412.2306, 2014.

KHAN, A.; SOHAIL, A.; ZAHOORA, U.; QURESHI, A. S. A survey of the recent architectures of deep convolutional neural networks. *CoRR*, abs/1901.06032, 2019.

KHOSLA, A.; SARMA, A. D.; HAMID, R. What makes an image popular? In: *Proceedings of the 23rd International Conference on World Wide Web*. New York, NY, USA: ACM, 2014. (WWW '14), p. 867–876. ISBN 978-1-4503-2744-2.

KIM, Y. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (Ed.). *Advances in Neural Information Processing Systems 25*. [S.l.]: Curran Associates, Inc., 2012. p. 1097–1105.

LARSSON, G.; MAIRE, M.; SHAKHNAROVICH, G. Fractalnet: Ultra-deep neural networks without residuals. *CoRR*, abs/1605.07648, 2016.

LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; JACKEL, L. D. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, MIT Press, Cambridge, MA, USA, v. 1, n. 4, p. 541–551, dez. 1989. ISSN 0899-7667.

LOPEZ, M. M.; KALITA, J. Deep learning applied to NLP. *CoRR*, abs/1703.03091, 2017.

LUONG, T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, 2015. abs/1508.04025, p. 1412–1421.

LYNCH, C.; ARYAFAR, K.; ATTENBERG, J. Images don't lie: Transferring deep visual semantic features to large-scale multimodal learning to rank. *CoRR*, abs/1511.06746, 2015.

MAATEN, L. van der; HINTON, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, v. 9, p. 2579–2605, 2008.

MAZLOOM, M.; HENDRIKS, B.; WORRING, M. Multimodal context-aware recommender for post popularity prediction in social media. In: *Proceedings of the on Thematic Workshops of ACM Multimedia 2017*. New York, NY, USA: ACM, 2017. (Thematic Workshops '17), p. 236–244. ISBN 978-1-4503-5416-5.

MAZLOOM, M.; RIETVELD, R.; RUDINAC, S.; WORRING, M.; DOLEN, W. van. Multimodal popularity prediction of brand-related social media posts. In: *Proceedings of the 24th ACM International Conference on Multimedia*. New York, NY, USA: ACM, 2016. (MM '16), p. 197–201. ISBN 978-1-4503-3603-1.

MCPARLANE, P. J.; MOSHFEGHI, Y.; JOSE, J. M. "nobody comes here anymore, it's too crowded"; predicting image popularity on flickr. In: *Proceedings of International Conference on Multimedia Retrieval*. New York, NY, USA: ACM, 2014. (ICMR '14), p. 385:385–385:391. ISBN 978-1-4503-2782-4.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

MNIH, V.; HEESS, N.; GRAVES, A.; KAVUKCUOGLU, K. Recurrent models of visual attention. In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. Cambridge, MA, USA: MIT Press, 2014. (NIPS'14), p. 2204–2212.

NAM, H.; HA, J.; KIM, J. Dual attention networks for multimodal reasoning and matching. *CoRR*, abs/1611.00471, 2016.

NOV, O.; NAAMAN, M.; YE, C. What drives content tagging: The case of photos on flickr. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, 2008. (CHI '08), p. 1097–1100. ISBN 978-1-60558-011-1.

OLAH, C. *Understanding LSTM Networks.* 2015.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1532–1543.

POSNER, M.; PETERSEN, S. The attention system of the human brain. *Annual review of neuroscience*, v. 13, p. 25–42, 02 1990.

RICHELDI, M.; ROSSOTTO, M. Class-driven statistical discretization of continuous attributes (extended abstract). In: *Proceedings of the 8th European Conference on Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 1995. (ECML'95), p. 335–338. ISBN 3-540-59286-5, 978-3-540-59286-0.

SANTOS, C. N. dos; GATTI, M. Deep convolutional neural networks for sentiment analysis of short texts. 2014.

SATHASIVAM, S.; ABDULLAH, W. A. T. W. Logic learning in hopfield networks. *CoRR*, abs/0804.4075, 2008.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

SUTSKEVER, I.; VINYALS, O.; LE, Q. V. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

SZEGEDY, C.; IOFFE, S.; VANHOUCKE, V.; ALEMI, A. A. Inception-v4, inception-resnet and the impact of residual connections on learning. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2017. (AAAI'17), p. 4278–4284.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. Going deeper with convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 1–9.

TAI, K. S.; SOCHER, R.; MANNING, C. D. Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075, 2015.

TURIAN, J.; RATINOV, L.; BENGIO, Y. Word representations: A simple and general method for semi-supervised learning. In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010. (ACL '10), p. 384–394.

WANG, C.; SUN, J.; CHEN, X. Feature fusion attention visual question answering. In: *Proceedings of the 2019 11th International Conference on Machine Learning and Computing*. New York, NY, USA: ACM, 2019. (ICMLC '19), p. 412–416. ISBN 978-1-4503-6600-7.

WANG, X.; LIU, Y.; SUN, C.; WANG, B.; WANG, X. Predicting polarities of tweets by composing word embeddings with long short-term memory. In: . [S.l.: s.n.], 2015. p. 1343–1353.

WELLS, W. M.; VIOLA, P.; ATSUMI, H.; NAKAJIMA, S.; KIKINIS, R. Multi-modal volume registration by maximization of mutual information. *Medical Image Analysis*, v. 1, n. 1, p. 35 – 51, 1996. ISSN 1361-8415.

Wong, A. K. C.; Chiu, D. K. Y. Synthesizing statistical knowledge from incomplete mixed-mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9, n. 6, p. 796–805, Nov 1987.

WU, B.; CHENG, W.-H.; ZHANG, Y.; HUANG, Q.; LI, J.; MEI, T. Sequential prediction of social media popularity with deep temporal context networks. In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2017. (IJCAI'17), p. 3062–3068. ISBN 978-0-9992411-0-3.

WU, B.; MEI, T.; CHENG, W.-H.; ZHANG, Y. Unfolding temporal dynamics: Predicting social media popularity using multi-scale temporal decomposition. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. [S.l.]: AAAI Press, 2016. (AAAI'16), p. 272–278.

XIE, Y.; LE, L.; ZHOU, Y.; RAGHAVAN, V. V. Deep learning for natural language processing. In: . [S.l.: s.n.], 2018.

XU, K.; BA, J.; KIROS, R.; CHO, K.; COURVILLE, A.; SALAKHUDINOV, R.; ZEMEL, R.; BENGIO, Y. Show, attend and tell: Neural image caption generation with visual attention. In: BACH, F.; BLEI, D. (Ed.). *Proceedings of the 32nd International Conference on Machine Learning*. Lille, France: PMLR, 2015. (Proceedings of Machine Learning Research, v. 37), p. 2048–2057.

YANDONG HAO ZONGBO, L. H. L. Survey of convolutional neural network. *Journal of Computer Applications*, Journal of Computer Applications, v. 36, n. 9, p. 2508, 2016.

YIN, W.; KANN, K.; YU, M.; SCHüTZE, H. Comparative study of cnn and rnn for natural language processing. *CoRR*, abs/1702.01923, 2017.

YOUNG, T.; HAZARIKA, D.; PORIA, S.; CAMBRIA, E. Recent trends in deep learning based natural language processing. *CoRR*, abs/1708.02709, 2017.

ZHAI, Z.; NGUYEN, D. Q.; VERSPOOR, K. Comparing CNN and LSTM character-level embeddings in bilstm-crf models for chemical and disease named entity recognition. *CoRR*, abs/1808.08450, 2018.

ZHANG, W.; WANG, W.; WANG, J.; ZHA, H. User-guided hierarchical attention network for multi-modal social image popularity prediction. In: *Proceedings of the 2018 World Wide Web Conference*. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018. (WWW '18), p. 1277–1286. ISBN 978-1-4503-5639-8.

ZHANG, X.; ZHAO, J. J.; LECUN, Y. Character-level convolutional networks for text classification. *CoRR*, abs/1509.01626, 2015.

ZHANG, Y.; WALLACE, B. C. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *CoRR*, abs/1510.03820, 2015.

ZHANG, Z.; CHEN, T.; ZHOU, Z.; LI, J.; LUO, J. How to become instagram famous: Post popularity prediction with dual-attention. *CoRR*, abs/1809.09314, 2018.