CENTRO UNIVERSITÁRIO FEI

MARIANA BASTOS DOS SANTOS

SUMARIZAÇÃO ABSTRATIVA DE TEXTO POR MEIO DO MODELO TEÓRICO COMPUTACIONAL COGNITIVO LIDA

São Bernardo do Campo

2022

MARIANA BASTOS DOS SANTOS

SUMARIZAÇÃO ABSTRATIVA DE TEXTO POR MEIO DO MODELO TEÓRICO COMPUTACIONAL COGNITIVO LIDA

Dissertação de mestrado, apresentada ao Centro Universitário da FEI para obtenção do título de Mestre em Engenharia Elétrica. Orientado pelo Prof. Dr. Paulo Sérgio Silva Rodrigues.

São Bernardo do Campo

2022

Bastos dos Santos, Mariana.
SUMARIZAÇÃO ABSTRATIVA DE TEXTO POR MEIO DO MODELO TEÓRICO COMPUTACIONAL COGNITIVO LIDA / Mariana Bastos dos Santos. São Bernardo do Campo, 2022.
114 p. : il.

Dissertação - Centro Universitário FEI.
Orientador: Prof. Dr. Paulo Sergio Silva Rodrigues.

1. Sumarização Abstrativa de Texto. 2. Modelos Computacionais Cognitivos. 3. PLN. 4. Seq2Seq. 5. LIDA. I. Sergio Silva Rodrigues, Paulo, orient. II. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

centro universitário FEI

APRESENTAÇÃO DE DISSERTAÇÃO
ATA DA BANCA EXAMINADORA

Programa de Pós-Graduação Stricto Sensu em Engenharia Elétrica

Mestrado

PGE-10

Aluno: Mariana Bastos dos Santos **Matrícula:** 119125-3

Título do Trabalho: Sumarização abstrativa de texto por meio do modelo teórico computacional cognitivo lida.

Área de Concentração: Processamento de Sinais e Imagens

Orientador: Prof. Dr. Paulo Sérgio Silva Rodrigues

Data da realização da defesa: 06/06/2022

ORIGINAL ASSINADA

Avaliação da Banca Examinadora:

A banca considerou que a aluna respondeu os questinamentos de maneira adequada e seu trabalho de Mestrado possui requisitos suficientes para ser considerada aprovada. No entanto, recomenda as correções que serão encaminhadas por e-mail.

São Bernardo do Campo, / / .

MEMBROS DA BANCA EXAMINADORA	
Prof. Dr. Paulo Sérgio Silva Rodrigues	Ass.: _____
Prof. Dr. Guilherme Alberto Wachs Lopes	Ass.: _____
Prof. Dr. Alexandre da Silva Simões	Ass.: _____

A Banca Julgadora acima-assinada atribuiu ao aluno o seguinte resultado:

APROVADO ☒ REPROVADO ☐

VERSÃO FINAL DA DISSERTAÇÃO

APROVO A VERSÃO FINAL DA DISSERTAÇÃO EM QUE FORAM INCLUÍDAS AS RECOMENDAÇÕES DA BANCA EXAMINADORA

_____

Aprovação do Coordenador do Programa de Pós-graduação

_____

Prof. Dr. Carlos Eduardo Thomaz

Aos meus pais, à minha irmã e à todas as pessoas que de alguma forma me ajudaram e acompanharam o desenvolvimento deste trabalho.

AGRADECIMENTOS

Dever cumprido. Ufa! Como foi difícil chegar aqui, e quão gratificante é poder escrever os agradecimentos. Esse trabalho me desafiou e aprofundou meu conhecimento de maneira significativa e estou muito honrada de ter podido finalizá-lo.

Nos próximos parágrafos quero dedicar aos que me ajudaram nesses 3 anos de desenvolvimento do trabalho.

Primeiramente, gostaria de agradecer ao Professor Paulo Sergio, meu orientador e a pessoa que me influenciou diretamente a me inscrever no Mestrado, quando eu não tinha segurança suficiente para isso. Muito obrigada pela confiança e sou eternamente grata por ter me apresentado tudo que sei sobre pesquisa e sobre a comunidade científica.

À minha mãe Nalva agradeço por todo o carinho e educação que foi me dado, pela sua dedicação diária e influência nas minhas conquistas, por estar sempre ao meu lado e por ser meu exemplo de mulher. Te amo muito!

Ao meu pai Antônio agradeço por sempre dedicar seu tempo e esforço para que eu possa realizar meus sonhos, e assim como minha mãe, pelo carinho e educação que foi me dado. Também te amo muito!

À minha irmã Vanessa agradeço por eu poder sempre dividir minhas angústias e felicidades, por me ensinar diariamente coisas que jamais esperava aprender com você e por ser essa pessoa com sede de conhecimento e de conquistas. Também te amo muito!

Aos meus amigos agradeço pelo suporte, pela troca de conhecimento e por terem transformado dias que podiam ser muito ruins em dias maravilhosos com a companhia de vocês.

Ao Centro Universitário da FEI agradeço a disponibilidade da infraestrutura e principalmente, por ter me tornado a profissional que sou hoje.

“Pela maior parte da história, "anônimo" foi uma mulher.”

Virginia Woolf

RESUMO

Os modelos de sumarização automática de texto surgiram na metade do século XX e por muito tempo foram desenvolvidos de maneira extrativa. Os modelos extrativos de sumarização de texto utilizam partes do texto original para construir o resumo (CELIKYILMAZ et al., 2018), gerando muitas vezes problemas de coerência e coesão quando as diferentes partes são lidas juntas. Em contrapartida, na última década, a abordagem abstrativa vem sendo bastante explorada, e diferentemente da extrativa, gera novas palavras que possivelmente não se encontram no texto original para construir o resumo (CELIKYILMAZ et al., 2018). Essa abordagem pode corrigir o problema de coerência e coesão, dado que se aproxima muito do modo como são construídos os resumos por humanos (SEE; LIU; MANNING, 2017). Porém, a sumarização abstrativa ainda enfrenta alguns problemas na geração do resumo, mesmo apresentando resultados satisfatórios em métricas automáticas de validação. Além disso, quando avaliados por humanos os resumos expõem problemas, como redundância, na dinâmica de leitura que ainda não é fluída. Por outro lado, há décadas são propostos modelos teóricos computacionais cognitivos que se baseiam nas teorias da psicologia e neurociência sobre a consciência, e que permitem a adaptação para diferentes aplicações, tendo ainda, como um dos modelos mais conhecidos, o LIDA (FRANKLIN et al., 2016). O presente trabalho propõe um modelo de sumarização abstrativa de texto baseado na estrutura teórica do LIDA utilizando técnicas já aplicadas para essa abordagem, tais como: *Sequence-to-Sequence* (Seq2Seq) (SUTSKEVER; VINYALS; LE, 2014), *Word2vec* (MIKOLOV et al., 2013a), *Long Short-Term Memory* (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) e Mecanismo de Atenção (BAHDANAU; CHO; BENGIO, 2014). Os resultados mostraram a importância dos módulos do LIDA na composição do modelo proposto, reforçando a importância dos módulos: Memória Perceptiva Associativa, Codeletes de Atenção e Espaço de Trabalho Global. Além disso, o trabalho ressaltou a fragilidade da métrica ROUGE na avaliação dos resumos gerados quanto a coerência e coesão. E por fim, a técnica de redução de dimensão utilizada no *word embedding*, se mostrou ineficaz para a tarefa.

Palavras-chave: Sumarização Abstrativa de Texto, Modelos Computacionais Cognitivos, LIDA, PLN, Redes Neurais, LSTM, Seq2Seq.

ABSTRACT

Automatic text summarization models appeared in the middle of the 20th century and for a long time were developed in an extractive manner. Extractive text summarization models use parts of the original text to construct the abstract (CELIKYILMAZ et al., 2018), which often generates coherence and cohesion problems when different parts are read together. On the other hand, in the last decade, an abstract approach has been extensively explored, and unlike the extractive one, it generates new words that possibly cannot be identified in the original text to construct the abstract (CELIKYILMAZ et al., 2018). This approach can solve the coherence and cohesion problem, whereas it is very close to the way that abstracts are constructed by humans (SEE; LIU; MANNING, 2017). However, the abstractive summarization still faces some problems in abstract generation, even presenting satisfactory results in automatic validation metrics, when evaluated by humans the abstracts expose problems in the reading dynamics which are not yet fluid. On the other hand, theoretical cognitive computacional models have been proposed for decades, which are based on psychology and neuroscience theories about consciousness and allow adaptation to different applications, having LIDA (FRANKLIN et al., 2016) as one of the best-known models. The present work proposes an abstractive text summarization model based on the LIDA theoretical structure using techniques already applied for the abstractive approach, such as: Sequence-to-Sequence (Seq2Seq) (SUTSKEVER; VINYALS; LE, 2014), Word2vec (MIKOLOV et al., 2013a), Long Short-Term Memory (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) and Attention Mechanism (BAHDANAU; CHO; BENGIO, 2014). The results showed the importance of LIDA modules in the composition of the proposed model, reinforcing the importance of the modules: Perceptual Associative Memory, Attention Codelettes and Global Workspace. In addition, the work highlighted the fragility of the ROUGE metric in the evaluation of the generated summaries regarding coherence and cohesion. And finally, the dimension reduction technique used in word embedding, proved to be ineffective for the task.

Keywords: Abstractive Text Summarization, Cognitive Computational Models, LIDA, NLP, Neural Networks, LSTM, Seq2Seq.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Com o rápido avanço tecnológico atual, a quantidade de informação disponível se dissemina rapidamente e a partir de múltiplas fontes, dificultando a organização e compreensão do conhecimento envolvido. Assim, ferramentas que direcionam o leitor ao conteúdo sem desvios e de maneira resumida, apenas com as informações-chaves em textos curtos, fazem-se necessárias. Sendo assim, a sumarização de texto automática, nesse contexto, ajuda na abstração dos principais pontos da informação num ambiente onde são compartilhadas diariamente uma infinidade de conteúdo, o que favorece à pesquisa e velocidade necessária na busca pelo conhecimento.

A sumarização automática de texto surgiu no fim dos anos 50, com o trabalho pioneiro de (LUHN, 1958) e por meio da abordagem extrativa, em que o resumo é composto por um subconjunto de frases ou palavras retiradas do texto principal. A maioria dos trabalhos ao longo dos anos foram desenvolvidos para gerar resumos extrativos (LUHN, 1958; EDMUNDSON, 1969; PAICE, 1990; KUPIEC; PEDERSEN; CHEN, 1995; JOHNSON et al., 1997; JING, 2000; KNIGHT; MARCU, 2002; MCDONALD, 2006; CLARKE; LAPATA, 2008; SAGGION; POIBEAU, 2013; RUSH; CHOPRA; WESTON, 2015; NALLAPATI; ZHAI; ZHOU, 2017). Devido a isso, muitos dos sistemas automáticos e atuais para sumarização de textos são construídos utilizando essa abordagem. Sobretudo, devido às sumarizações resultantes desses sistemas serem combinações de palavras e frases já existentes no texto, a leitura e compreensão do leitor pode ser limitada devido a problemas de coerência e coesão do resumo. Sendo assim, com o objetivo de aproximar as sumarizações geradas de forma automática aos resumos gerados por humanos, surgiu a sumarização abstrativa de texto.

A partir do trabalho desenvolvido por (SUTSKEVER; VINYALS; LE, 2014), com uma aplicação bem sucedida da arquitetura *sequence-to-sequence*, a sumarização abstrativa começou a ser majoritariamente explorada e desenvolvida. Porém, essa abordagem possui algumas irregularidades quando os resultados são avaliados por seres humanos, diferentemente do que ocorre quando avaliados por métricas automáticas, ou seja, a leitura do resumo muitas vezes não é fluída e pode ser de difícil compreensão. Essas irregularidades são hoje estudadas pelos trabalhos no estado da arte, demandando ainda espaço para melhorias.

Por outro lado, áreas como neurociência, psicologia, filosofia e inteligência artificial começaram a desenvolver pesquisas sobre a ciência cognitiva. Inspiradas pela doutrina do behaviorismo ou comportamentalismo no início do século XX, a filosofia e a psicologia buscavam

entender, por meio de estudos, o funcionamento da mente e seus processos. Em outro momento do século XX, a neurociência desenvolvia pesquisas voltadas ao funcionamento dos neurônios focada em processos biológicos. E influenciada por todas essas áreas, a inteligência artificial surgiu com o objetivo de representar os processos, até então conhecidos da mente, por meio de regras lógico computacionais.

Com a expectativa de reagir a novas e problemáticas situações de uma forma mais flexível e humana do que os tradicionais sistemas de Inteligência Artificial, surgem os modelos computacionais cognitivos. De um modo geral, são modelos baseados em teorias da consciência, capazes de resolver problemas tendo como entrada percepções do ambiente externo, sendo então compreendidas e influenciadas por uma gama de muitas outras percepções já então conhecidas pelo modelo. Dentre os modelos desenvolvidos o LIDA (FRANKLIN et al., 2016) é um dos mais conhecidos.

O *Learning Intelligent Decision Agent* (LIDA) é um modelo teórico computacional cognitivo construído como um agente baseado no processo de cognição humana dividido em 3 grandes fases: a primeira de percepção e entendimento, a segunda de atenção e a terceira de ação e aprendizado. Cada fase é composta por um conjunto de módulos, os quais possuem determinadas tarefas que constituem o modelo. Por meio das tarefas que são executadas em cada um dos módulos do LIDA, é possível determinar os passos necessários para uma tarefa maior ser executada, como por exemplo, a tarefa de sumarização de texto.

Considerando que o processo de leitura e compreensão de texto está estreitamente ligado à processos cognitivos, e que a literatura científica ainda não apresentou uma ampla abordagem para modelagem de textos baseados nesses modelos cognitivos, o trabalho apresentado aqui expõe uma nova proposta para estudo da sumarização de textos modeladas como um sistema cognitivo baseado no LIDA. A ideia geral da proposta é utilizar estratégias consolidadas de sumarização abstrativa suportadas por modelos cognitivo também consolidados.

1.1 OBJETIVO

O objetivo do presente trabalho consiste em desenvolver um modelo de sumarização abstrativa de texto utilizando como base a estrutura teórica do LIDA.

1.2 ESTRUTURA DO TRABALHO

As estruturas dos demais capítulos deste trabalho serão apresentadas nos parágrafos a seguir.

No Capítulo 2 serão apresentados os principais trabalhos no estado da arte sobre o tema desenvolvido neste trabalho.

No Capítulo 3 serão esclarecidos as principais técnicas e teorias levantadas como conceitos fundamentais para a evolução do trabalho.

No Capítulo 4 será descrita a estrutura de desenvolvimento da proposta para atingir o objetivo deste trabalho.

No Capítulo 5 serão analisados e discutidos os resultados obtidos dos experimentos realizados.

E por fim, no Capítulo 6 são apresentados: os pontos levantados nas análises dos resultados, a discussão de como os resultados podem ser incrementados, a contribuição do trabalho e os trabalhos futuros.

2 TRABALHOS RELACIONADOS

A sumarização automática de texto é uma tarefa estudada desde o final do século XX (KUPIEC; PEDERSEN; CHEN, 1995), sendo principalmente, naquela época, desenvolvida por meio da abordagem extrativa, a qual utiliza as sentenças principais de um determinado texto para criar o resumo. A abordagem extrativa evoluiu, e atualmente são aplicados muitos dos modelos principais de *Deep Learning*, sendo possível observar na Tabela 1. Em paralelo, enquanto a abordagem extrativa era aprimorada, surgia a abordagem abstrativa (BANKO; MITTAL; WITBROCK, 2000; ZAJIC; DORR; SCHWARTZ, 2004; COHN; LAPATA, 2008; WOODSEND; FENG; LAPATA, 2010), na qual o resumo é gerado utilizando novas sentenças ou sentenças reformuladas, o que exige mais das ferramentas de *Deep Learning*, pois tem a finalidade de tornar o processo de sumarização automática mais próximo do exercido pelo ser humano.

Alguns trabalhos do estado da arte de sumarização abstrativa, como pode ser visto na Tabela 2, utilizam técnicas bio-inspiradas como: redes neurais recorrentes (RNN) de (RUMELHART; HINTON; WILLIAMS, 1986), memória de curto-longo prazo (LSTM) de (HOCHREITER; SCHMIDHUBER, 1997) e o mecanismo de atenção de (BAHDANAU; CHO; BENGIO, 2014). Essas técnicas podem ser justificadas, pois aproximam a sumarização automática da tarefa exercida pelo ser humano, como citado anteriormente.

Por outro lado, arquiteturas cognitivas foram desenvolvidas ao longo dos anos (SUN, 1997; FRANKLIN; KELEMEN; MCCAULEY, 1998; SUN, 1999; JOHNSON; CAULFIELD; TAYLOR, 2000; FRANKLIN, 2003; TAYLOR, 2007; STARZYK; PRASAD, 2011; FRANKLIN et al., 2014, 2016; KHAYI; FRANKLIN, 2018; MCCALL et al., 2020) com a expectativa de executar tarefas de maneira mais humana do que tradicionais modelos de inteligência artificial. De um modo geral, são modelos computacionais baseados no funcionamento cognitivo a partir de estudos sobre a teoria da consciência das áreas de psicologia e neurociência.

Tendo em vista o estado da arte da sumarização abstrativa e que uma arquitetura cognitiva possua módulos de memórias de curto e longo prazo, e de atenção, por exemplo, é justicável que uma conexão entre a tarefa de sumarização abstrativa de texto e uma arquitetura cognitiva seja feita. Sendo assim, são apresentados neste capítulo os trabalhos mais recentes de sumarização de texto e os trabalhos relacionados a arquiteturas cognitivas, para que seja possível construir essa ligação entre a tarefa de sumarização e sua execução em uma arquitetura cognitiva.

Na Seção 2.1 são abordados, inicialmente, os trabalhos mais recentes da abordagem de sumarização extrativa.

Na Seção 2.2, são apresentados os trabalhos mais recentes da abordagem de sumarização abstrativa.

Na Seção 2.3, são expostos alguns trabalhos que utlizam abordagem híbrida - extrativa e abstrativa juntas.

Na sequência, na Seção 2.4 são apresentados os trabalhos de revisões de sumarização de texto.

Por fim, na Seção 2.5, são apresentados os trabalhos de modelos computacionais cognitivos.

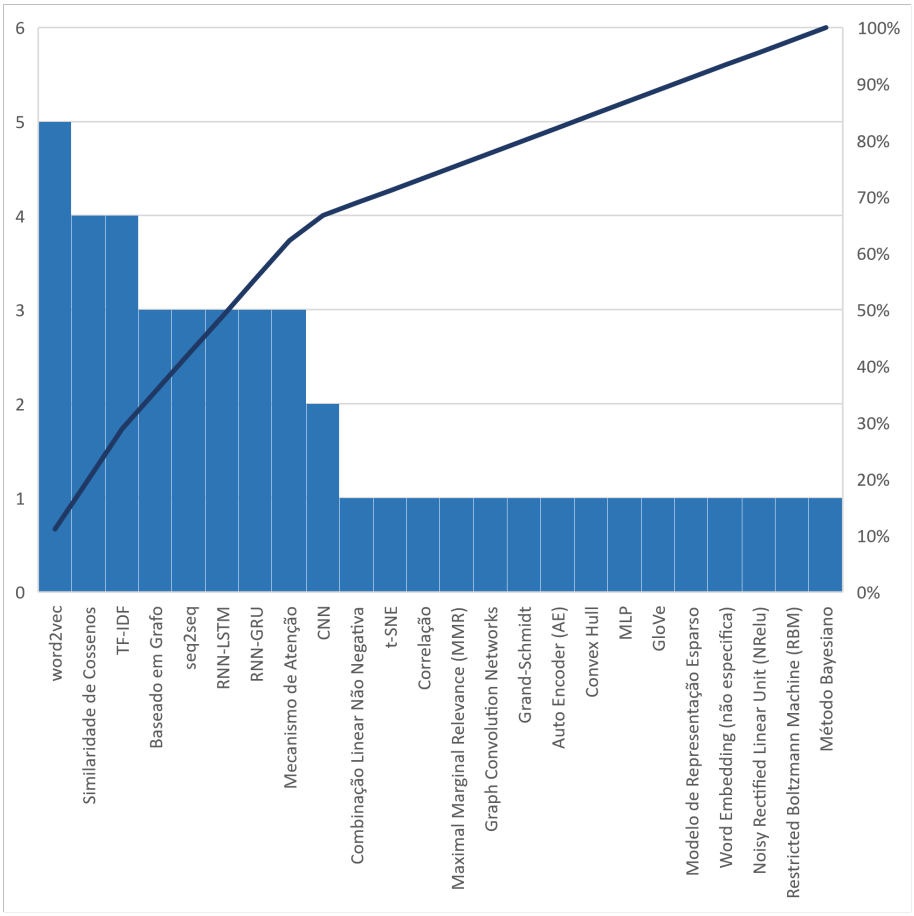
2.1 SUMARIZAÇÃO EXTRATIVA

Os trabalhos de sumarização extrativa recentes trabalham com técnicas de *Deep Learning* e apresentam os melhores resultados nas principais métricas automáticas de avaliação de sumarização utilizadas, ROUGE-1, ROUGE-2 e ROUGE-L de (LIN, 2004), como observado na Tabela 1. Também é possível avaliar os conjuntos de dados mais utilizados nesses trabalhos e ainda, através da Figura 1 é possível ter o conhecimento de quais métodos são mais utilizados pelos trabalhos apresentados nessa seção.

Tabela 1 – Síntese dos trabalhos de Sumarização Extrativa apresentados ao longo dessa Seção 2.1.

Trabalhos	**Métodos Utilizados**	Banco de Dados	ROUGE-1	ROUGE-2	ROUGE-L
(CHENG; LAPATA, 2016)	Mecanismo de Atenção + CNN + RNN-LSTM + word2vec + Seq2Seq	CNN/Daily Mail	56	24,9	50,2
(NARAYAN et al., 2017)	RNN-LSTM + CNN + Mecanismo de Atenção + word2vec + Seq2Seq	CNN/Daily Mail	54,2	21,6	48,1
(ZHOU et al., 2018)	RNN-GRU + MLP + GloVe	CNN/Daily Mail	41,59	19,01	37,98
(NALLAPATI; ZHAI; ZHOU, 2017)	RNN-GRU + word2vec	CNN/Daily Mail	39,6	16,2	35,3
(NALLAPATI; ZHAI; ZHOU, 2017)	RNN-GRU + word2vec	DUC (2002)	46,6	23,1	43,03
(CHENG; LAPATA, 2016)	Mecanismo de Atenção + CNN + RNN-LSTM + word2vec + Seq2Seq	DUC (2002)	47,4	23	43,5
(BARRIOS et al., 2016)	Baseado em Grafo	DUC (2002)	40,42	18,3	-
(ROSSIELLO; BASILE; SEMERARO, 2017)	word2vec + TF-IDF + Similaridade de Cossenos + t-SNE	DUC (2004)	38,81	9,97	-
(YASUNAGA et al., 2017)	Baseado em Grafo + RNN-GRU + word2vec + Graph Convolution Networks	DUC (2004)	38,23	9,48	-
(COHAN; GOHARIAN, 2017)	Baseado em Grafo + Similaridade de Cossenos + TF-IDF	TAC (2004)	45	14	42
(YOGATAMA; LIU; SMITH, 2015)	Maximal Marginal Relevance (MMR) + convex hull + Grand-Schmidt	TAC (2008)	37,5	9,58	-
(YOGATAMA; LIU; SMITH, 2015)	Maximal Marginal Relevance (MMR) + convex hull + Grand-Schmidt	TAC (2009)	34,37	8,76	-
(MA; SUN, 2017)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção + Word Embedding (não especifica) + Similaridade de Cossenos	LCSTS	33,1	20	30,1
(LIU; YU; DENG, 2015)	Modelo de Representação Esparso + Correlação + Combinação Linear Não Negativa	DUC(2006)	34,44	5,12	-
(MORADI; GHADIRI, 2018)	Método Bayeasiano + TF-IDF	Artigos Biomédicos	78,86	35,29	-
(YOUSEFI-AZAR; HAMEY, 2017)	TF-IDF + Auto Encoder + RBM + nrelu + Similaridade de Cossenos	SKE	51,1	-	-

Figura 1 – Gráfico que representa o número de trabalhos por métodos utilizados no desenvolvimento de modelos de sumarização extrativa de texto, sendo as barras o número de trabalhos e a linha o percentual de trabalhos acumulado. Nota-se que os 7 primeiros métodos são utilizados em 60% dos trabalhos.



Fonte: autora

2.1.1 Modelos de Sumarização Extrativa

(LIU; YU; DENG, 2015) abordaram o problema de sumarização por extração de sentenças de múltiplos documentos aplicando códigos esparsos que representam cada sentença como uma combinação linear não negativa das sentenças resumidas e apresentaram um novo *framework* para esse problema. O *framework* foi construído por meio de representação esparsa de dois níveis para retratar o processo de sumarização de múltiplos documentos. Exaustivos experimentos em conjuntos de dados mostraram que o *framework* proposto é bastante eficaz.

(CHENG; LAPATA, 2016) propuseram uma abordagem orientada a dados com base em *features* contínuas de sentenças. Para isso, desenvolveram um *framework* geral de sumarização de documento único que pode ser utilizado para extrair sentenças ou palavras. O *framework*

inclui um leitor ou codificador hierárquico de documentos baseado em rede neural, além de possuir um extrator de conteúdo baseado em atenção. Os resultados experimentais, em dois conjuntos de dados para sumarização, demonstraram que o *framework* obteve resultados comparáveis ao estado da arte.

(MA; SUN, 2017) buscaram melhorar a relevância semântica entre textos fontes e textos resumidos. Para isso, apresentaram um modelo neural baseado em relevância semântica para estimular uma alta similaridade semântica entre textos e resumos, ou seja, no modelo, o texto fonte é representado por um codificador fechado de atenção, enquanto a representação do resumo é produzida pelo decodificador. Os experimentos realizados mostraram que o modelo proposto performou melhor que os sistemas do estado da arte nos conjuntos de dados *benchmark*.

(NALLAPATI; ZHAI; ZHOU, 2017) apresentaram o SummaRuNNer, um modelo de sequência recorrente baseado em rede neural para sumarização de documentos por extração. Os autores trataram a sumarização por extração como um problema de classificação de sequência em que, cada sentença é visitada sequencialmente na ordem do documento original e uma decisão binária é tomada se deve ou não ser incluída no resumo. Os experimentos mostraram que o modelo tem melhor desempenho do que ou é comparável aos modelos de aprendizado profundo de última geração.

(NARAYAN et al., 2017) exploraram informações secundárias no contexto de sumarização extrativa para documento único. Sendo assim, desenvolveram um *framework* para sumarização de documento único, composto por um codificador hierárquico de documentos e um extrator baseado em atenção voltado para informações secundárias. Em experimentos realizados mostraram que, a sumarização extrativa com informações secundárias, supera consistentemente as abordagens que não utilizam nenhuma dessas informações, tanto em termos de informatividade quanto de fluência.

(YASUNAGA et al., 2017) propuseram um sistema neural de sumarização de múltiplos documentos, utilizando uma rede convolucional baseada em grafos com *embedding* de sentenças obtidas de redes neurais recorrentes como *features* dos nós de entrada. O modelo proposto aprimorou as abordagens extrativas tradicionais baseadas em grafos e em modelo de sequência GRU sem grafo, alcançando resultados competitivos em relação a outros sistemas de resumo de múltiplos documentos de última geração.

(ZHOU et al., 2018) apresentaram um novo *framework* de rede neural *end-to-end* para sumarização extrativa de documentos, aprendendo a pontuar e selecionar sentenças de forma

conjunta. Para isso, o *framework* lê inicialmente as frases do documento de entrada com um codificador hierárquico para obter a representação das frases. Em seguida, cria o resumo da saída extraindo sentenças uma a uma. A abordagem proposta integra a estratégia de seleção ao modelo de pontuação, que prediz diretamente a importância relativa dada as frases selecionadas anteriormente. Experimentos no conjunto de dados *CNN/Daily Mail* mostraram que, o *framework* proposto supera significativamente os modelos de resumo extrativo do estado da arte.

(YOGATAMA; LIU; SMITH, 2015) propuseram um método de sumarização extrativa baseado em maximizar o volume em um espaço vetorial semântico (conjunto de sentenças que constroem um sentido ou a semântica das palavras). Para isso, construíram um *embedding* para cada sentença em um espaço semântico e obtiveram a sumarização a partir da escolha de um subconjunto de sentenças de volume maximizado pelo *convex hull* nesse espaço. Sendo assim, utilizaram um algoritmo guloso, baseado no processo de *Gram-Schmidt* (LAPLACE, 1820), para executar com eficiência a maximização de volume. O método proposto superou as abordagens de resumo de última geração em conjuntos de dados de referência.

(COHAN; GOHARIAN, 2017) propuseram uma abordagem para resumo de artigos científicos utilizando o contexto de citação - trecho do artigo referência que reflete a citação - e o modelo de discurso do documento. Sendo assim, extraíram o contexto de citação dos artigos de referência para cada citação. Então, utilizando as características discursivas das citações, bem como a estrutura em comum dos contextos de citação, extraíram sentenças candidatas ao resumo. O resumo final é formado da maximização do grau de informatividade e singularidade da sentença no resumo. O método proposto mostrou melhoras efetivas sobre abordagens existentes de sumarização, melhoria superior a 30% em relação ao melhor desempenho das referências.

(ROSSIELLO; BASILE; SEMERARO, 2017) propuseram um método baseado em centróide de sumarização de texto que explora a capacidade de estruturação dos *word embeddings* - representação vetorial contínua capaz de capturar a informação sintática e semântica de uma palavra. Para isso, adaptaram o método baseado em centróide introduzindo uma representação distribuída das palavras em que cada palavra em um documento é representada por um vetor de números reais de um tamanho estabelecido, tirando vantagem das propriedades estruturais dos *word embeddings*. Os resultados comprovaram a eficácia da representação vetorial contínua das palavras em comparação ao modelo *bag-of-words*.

(YOUSEFI-AZAR; HAMEY, 2017) apresentaram um método de resumo de documento único, orientado a consultas extrativas, e um codificador automático (AE) - uma rede *feed-forward* com a função de aprender para reconstruir uma entrada x - profundo para calcular o espaço de *feature* de entrada do termo de maior frequência. Inicialmente, desenvolveram uma representação de palavras local na qual cada vocabulário é criado para construir as representações de sentença de entrada no documento analisado. Na sequência, adicionaram ruídos aleatórios no vetor de representação de palavras, afetando ambos entrada e saída do codificador automático. Em um outro momento, após ranquear as sentenças do documento baseado na similaridade de cossenos, as mesmas devem ser selecionadas para gerar a sumarização. Por fim, a sumarização final é obtida selecionando as sentenças que ocorrem frequentemente nas sumarizações individuais. Experimentos mostraram que o codificador automático, utilizando variáveis locais, proporciona claramente um espaço de *feature* mais discriminativo e melhora o *recall* em 11,2% na média.

(MORADI; GHADIRI, 2018) descreveram um método Bayeasiano de sumarização de documentos de textos biomédicos. O sintetizador Bayesiano inicialmente mapeia o texto de entrada para os conceitos do Sistema Unificado de Linguagem Médica (UMLS) (NELSON et al., 2010). Em seguida, seleciona os mais importantes para serem utlizados como *features* de classificação. Os resultados mostraram que ao utilizar o método de seleção de *features* não utilizando a frequência bruta, o sintetizador Bayesiano supera os sintetizadores biomédicos que dependem da frequência de conceitos, independentes de domínio e métodos *baseline*.

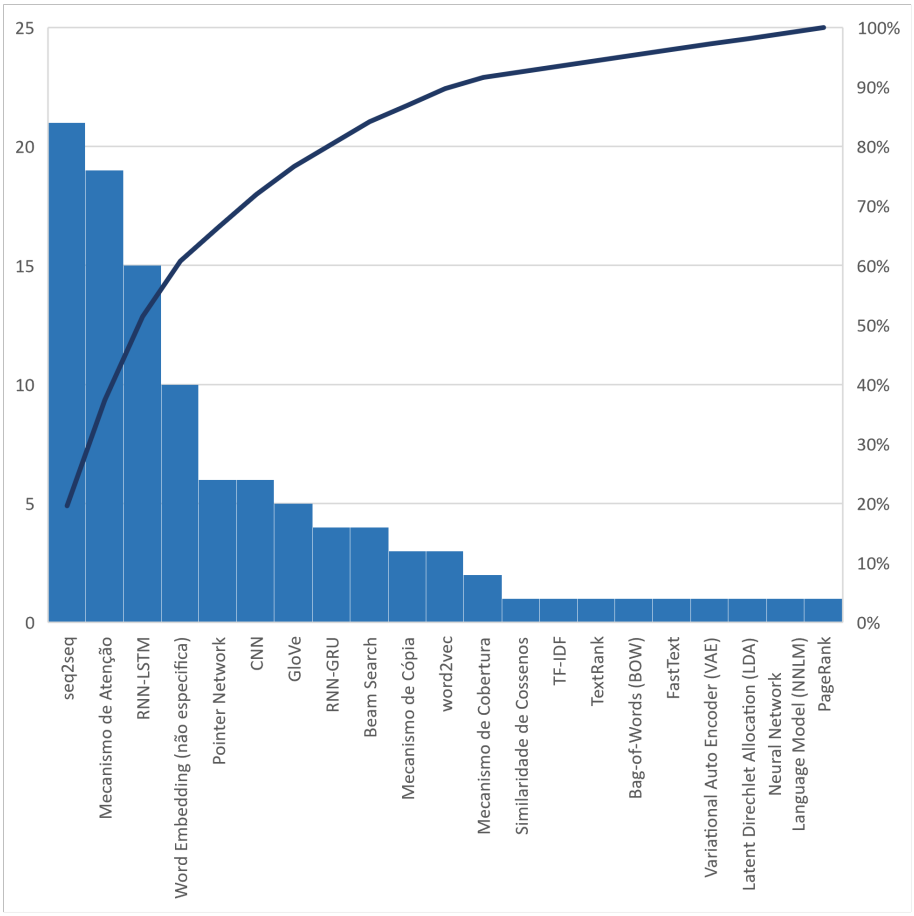
(BARRIOS et al., 2016) apresentaram novas alternativas à função de similaridade para o algoritmo *TextRank* em sumarização automática de texto. Para isso, se basearam em ideias para mudar a maneira que as distâncias entre as sentenças são computadas para ponderar os pesos das arestas do grafo usado para o *PageRank*. Essas medidas de similaridade são ortogonais ao modelo *TextRank*, portanto, podem ser facilmente integradas ao algoritmo. Foram encontradas tais variações para produzir significativas melhorias sobre o algoritmo original. O *TextRank* performou 2,84% a mais que o *baseline*, tendo então uma melhora de 2,92% acima da pontuação do *TextRank*, sendo um importante resultado.

2.2 SUMARIZAÇÃO ABSTRATIVA

Os trabalhos de sumarização abstrativa, assim como os da abordagem extrativa, apresentam melhores resultados nos modelos de *Deep Learning*, nas mesmas métricas avaliadas

ROUGE-1, ROUGE-2 e ROUGE-L como visto na Tabela 2. Os conjuntos de dados utilizados por esses trabalhos também são expostos na Tabela 2. Por fim, através da Figura 2 é possível ter o conhecimento de quais métodos são mais utilizados pelos trabalhos apresentados nessa seção.

Figura 2 – Gráfico que representa o número de trabalhos por métodos utilizados no desenvolvimento de modelos de sumarização abstrativa de texto, sendo as barras o número de trabalhos e a linha o percentual acumulado de trabalhos. Nota-se que os 3 primeiros métodos são utilizados por 60% dos trabalhos.



Fonte: autora

Tabela 2 – Síntese dos trabalhos de Sumarização Abstrativa apresentados ao longo dessa Seção 2.2

Trabalhos	**Métodos Utilizados**	Banco de Dados	ROUGE-1	ROUGE-2	ROUGE-L
(PAULUS; XIONG; SOCHER, 2017)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção + Pointer Network + GloVe	CNN/Daily Mail	42,16	15,75	39,08
(CELIKYILMAZ et al., 2018)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção + Pointer Network + GloVe	CNN/Daily Mail	41,69	19,47	37,92
(GEHRMANN; DENG; RUSH, 2018)	Seq2Seq + Mecanismo de Atenção + Mecanismo de Cópia + RNN-LSTM + GloVe	CNN/Daily Mail	41,22	18,68	38,34
(CHEN; BANSAL, 2018)	Seq2Seq + RNN-LSTM + Pointer Network + word2vec + Mecanismo de Cópia + Mecanismo de Atenção	CNN/Daily Mail	40,88	17,8	38,54
(KRYŚCIŃSKI et al., 2018)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção + Word Embedding (não especifica)	CNN/Daily Mail	40,19	17,34	37,52
(GUO; PASUNURU; BANSAL, 2018)	Seq2Seq + Mecanismo de Atenção + RNN-LSTM + Pointer Network + Mecanismo de Cobertura + Word Embedding (não especifica)	CNN/Daily Mail	39,84	17,63	36,54
(SEE; LIU; MANNING, 2017)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção + Pointer Network + Mecanismo de Cobertura + GloVe	CNN/Daily Mail	39,53	17,28	36,38
(LI, C. et al., 2018)	Seq2Seq + Mecanismo de Atenção + textrank + Pointer Network + RNN-LSTM + Word Embedding (não especifica)	CNN/Daily Mail	38,95	17,12	35,68
(TAN; WAN; XIAO, 2017)	Seq2Seq + Word Embedding (não especifica) + Mecanismo de Atenção + RNN-LSTM + PageRank	CNN/Daily Mail	38,1	13,9	34,9
(NALLAPATI et al., 2016)	Seq2Seq + RNN-GRU + Mecanismo de Atenção + word2vec + TF-IDF	CNN/Daily Mail	35,46	13,3	32,65
(SONG; HUANG; RUAN, 2019)	Seq2Seq + RNN-LSTM + CNN + Word Embedding (não especifica) + Mecanismo de Atenção	CNN/Daily Mail	34,9	17,8	-
(CAO et al., 2018)	CNN + Similaridade de Cossenos + word2vec	DUC (2004)	38,27	9,66	-
(LI, P. et al., 2017)	Seq2Seq + RNN-GRU + VAE + Word Embedding (não especifica) + Beam Search	DUC (2004)	31,79	10,75	27,48
(WANG et al., 2018)	Seq2Seq + Word Embedding (não especifica) + CNN + LDA + Mecanismo de Atenção	DUC (2004)	31,15	10,85	27,68
(LI; BING; LAM, 2018)	Seq2Seq + RNN-GRU + Mecanismo de Atenção + Word Embedding (não especifica) + Beam Search	DUC (2004)	29,41	9,84	25,85
(CHOPRA; AULI; RUSH, 2016)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção	DUC (2004)	28,97	8,26	24,06
(RUSH; CHOPRA; WESTON, 2015)	Seq2Seq + Word Embedding (não especifica) + NNLM + BOW + CNN + Mecanismo de Atenção	DUC (2004)	28,18	8,49	23,41
(WANG et al., 2018)	Seq2Seq + Word Embedding (não especifica) + CNN + LDA + Mecanismo de Atenção	Gigaword	36,92	18,29	34,58
(LIN et al., 2018)	Seq2Seq + CNN + Word Embedding (não especifica) + RNN-LSTM	Gigaword	36,3	18	33,8
(LI, P. et al., 2017)	Seq2Seq + RNN-GRU + VAE + Word Embedding (não especifica) + Beam Search	Gigaword	36,27	17,57	33,62
(LI; BING; LAM, 2018)	Seq2Seq + RNN-GRU + Mecanismo de Atenção + Word Embedding (não especifica) + Beam Search	Gigaword	36,05	17,25	33,49
(GUO; PASUNURU; BANSAL, 2018)	Seq2Seq + Mecanismo de Atenção + RNN-LSTM + Pointer Network + Mecanismo de Cobertura + Word Embedding (não especifica)	Gigaword	35,98	17,76	33,63
(SONG; ZHAO; LIU, 2018)	Seq2Seq + RNN-LSTM + Mecanismo de Cópia + Mecanismo de Atenção + Beam Search	Gigaword	35,47	17,66	33,52
(CHOPRA; AULI; RUSH, 2016)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção	Gigaword	33,78	15,97	31,15
(RUSH; CHOPRA; WESTON, 2015)	Seq2Seq + Word Embedding (não especifica) + NNLM + BOW + CNN + Mecanismo de Atenção	Gigaword	31	12,65	28,34
(NEMA et al., 2017)	Seq2Seq + RNN-GRU + Mecanismo de Atenção + RNN-LSTM + GloVe	Depatepedia	41,26	18,75	40,43
(LI; BING; LAM, 2018)	Seq2Seq + RNN-GRU + Mecanismo de Atenção + Word Embedding (não especifica) + Beam Search	LCTS	37,51	24,68	35,02
(MA et al., 2017)	Seq2Seq + RNN-LSTM + Mecanismo de Atenção	LCSTS	33,3	20	30,1
(KIM; KIM; KIM, 2018)	Seq2Seq + FastText + Mecanismo de Atenção + CNN	TIFU-Short	20,2	7,4	19,8

(NALLAPATI et al., 2016) propuseram novos modelos que abordam problemas críticos na sumarização que não são modelados adequadamente pela arquitetura básica. Os autores modelaram a sumarização abstrativa utilizando redes neurais recorrentes *encoder-decoder*, baseadas em atenção para modelar palavras-chave, capturar hierarquia de uma sentença para uma palavra estruturada e emitir palavras que são raras ou não ocorrem no treinamento. Os experimentos apresentaram resultados promissores, sendo que em dois diferentes conjuntos de dados obtiveram performance acima do estado da arte.

(LI, P. et al., 2017) propuseram um novo *framework* para sumarização abstrativa de texto, baseado na arquitetura *sequence-to-sequence* (SUTSKEVER; VINYALS; LE, 2014) - é aplicada quando as sequências de entrada e saída são de tamanhos diferentes, sendo composta de um codificador que recebe os dados da sequência de entrada e de um decodificador que gera a sequência de saída - , equipado com um componente de modelagem de estrutura latente. Os autores desenvolveram Autoencoders Variacionais (VAEs) (KINGMA; WELLING, 2013; REZENDE; MOHAMED; WIERSTRA, 2014) - modelo generativo direcionado com variáveis latentes - como o modelo base para o *framework* generativo, o qual consegue lidar com o problema de inferência associado à modelagem sequencial generativa. Além disso, foram adicionadas dependências históricas no decodificador generativo recorrente profundo para a modelagem da estrutura latente. Na sequência, o decodificador determinístico discriminativo padrão e o decodificador generativo recorrente são integrados num *framework* decodificador unificado. As sumarizações alvo foram decodificadas baseadas tanto nas variáveis determinísticas discriminativas quanto na informação estrutural latente generativa. Todos os parâmetros neurais foram aprendidos por *back-propagation* no paradigma *end-to-end*. Experimentos em bases de dados *benchmark* mostraram melhora nos resultados estando acima dos métodos do estado da arte.

(MA et al., 2017) buscaram melhorar a relevância semântica entre o texto fonte e os resumos para a sumarização de redes sociais chinesas. No modelo proposto, o texto de origem foi representado por uma porta codificadora de atenção (HAHN; KELLER, 2016) - responsável por mensurar a importância de uma palavra, gerando como saída um *score* de importância - , enquanto a representação da síntese foi produzida por um decodificador. Além disso, o *score* de similaridade entre a representação foi maximizado durante o treinamento. Os experimentos mostraram que o modelo proposto superou os sistemas de referência em um conjunto de dados de rede social.

(NEMA et al., 2017) propuseram um modelo para resumir com base em consulta e no paradigma *encode-attend-decode* ou mecanismo de atenção - modelo que primeiro computa uma representação vetorial para um documento e na sequência produz uma sumarização contextual para uma palavra de cada vez. Para isso, utilizaram um modelo de atenção voltado à consulta - além do modelo de atenção ao documento - que aprende a se concentrar em diferentes partes da consulta e diferentes momentos, ao invés de utilizar uma representação estática para a consulta. Além disso, utilizaram um novo modelo de atenção baseado na diversidade que visa melhorar o problema de repetir frases no resumo. Experimentos mostraram que o modelo proposto superou, com facilidade, os modelos de *encode-attend-decode* com um ganho de 28% (absoluto) na métrica ROUGE-L.

(SEE; LIU; MANNING, 2017) propuseram uma nova arquitetura a qual amplia o modelo baseado em atenção, *sequence-to-sequence*, de duas formas ortogonais. Para isso, utilizaram inicialmente, uma rede híbrida *pointer-generator* a qual pode copiar palavras do texto via *pointer*, ajudando na reprodução precisa de informações e mantendo a capacidade de produzir novas palavras através do *generator*. Na sequência, utilizaram a técnica *coverage* para observar o que foi resumido, minimizando as repetições. O modelo apresentou muitas capacidades abstrativas, reduziu imprecisões e repetição, e também performou significativamente acima do resultado do estado da arte.

(PAULUS; XIONG; SOCHER, 2017) introduziram um modelo de rede neural, para codificar uma sentença de entrada em um vetor fixo e criar uma nova sequência de saída desse vetor, com uma nova atenção e um novo método de treinamento. Para isso, apresentaram um mecanismo chave de atenção e um novo objetivo de aprendizado para resolver o problema de frase repetida. Os autores utilizaram uma atenção intra-temporal no codificador, que registra pesos de atenção anteriores para cada um dos tokens de entrada, enquanto um modelo sequencial de atenção interna no decodificador leva em consideração quais palavras já foram geradas pelo decodificador. O modelo obteve 41,16 na pontuação ROUGE-1 no conjunto de dados *CNN/-Daily Mail* (HERMANN et al., 2015), uma melhoria em relação aos modelos do estado da arte anteriores.

(CHEN; BANSAL, 2018) propuseram um modelo de resumo rápido e preciso, que primeiro seleciona sentenças importantes e depois reescreve de forma abstrativa, para gerar um resumo geral conciso. O modelo proposto, uma arquitetura híbrida extrativa-abstrativa com aprendizado por reforço baseado em política, primeiro utiliza um agente extrator para selecionar sentenças importantes ou em destaque, e então aplica uma rede abstrativa para reescrever

cada uma das sentenças extraídas. Para solucionar o comportamento não diferencial do extrator e treinar em pares de documentos, os autores empregam o gradiente político ator-crítico com uma métrica de recompensas no nível de sentenças para conectar as duas redes neurais, extrativa e abstrativa, e para aprender sentenças importantes. O modelo alcançou o novo estado da arte no conjunto de dados *CNN/Daily Mail* (HERMANN et al., 2015), além de uma melhor generalização no conjunto de dados DUC (2002) (STANDARDS; (NIST), 2002).

(KIM; KIM; KIM, 2018) resolveram o problema de sumarização abstrativa propondo um novo modelo, nomeado Redes de Memória Multinível (MMN) que é equipado com uma memória multinível para armazenar as informações de texto de diferentes níveis de abstração. Resultados mostraram que o MMN superou os modelos de sumarização no estado da arte.

(LI, C. et al., 2018) propuseram um modelo orientado para geração de resumo abstrativo, combinando o método extrativo e abstrativo de sumarização. Primeiramente, obtiveram palavras-chave do texto por meio do método extrativo. Em seguida, apresentaram uma rede guia de informações importantes que codifica as palavras-chave para representação de informações importantes, o qual orienta o processo de geração. Além disso, utilizaram um mecanismo guia de predição, obtendo o valor a longo prazo para decodificação futura, orientando, ainda mais, a geração de resumo. Os experimentos mostraram que o modelo proposto conduziu melhorias significativas.

Para resolver os problemas de repetição e irrelevância semântica dos modelos *sequence-to-sequence*, (LIN et al., 2018) propuseram um *framework* de codificação global, que controla o fluxo de informações do codificador para o decodificador com base nas informações globais do contexto de origem. Para isso, definiram uma porta convolucional para executar o codificador global do contexto de origem. A porta, baseada em rede neural convolutiva, filtra cada saída do codificador com base no contexto global devido ao compartilhamento de parâmetros, para que as representações em cada etapa sejam refinadas com a consideração do contexto global. Os experimentos, nos conjuntos de dados LCSTS (HU; CHEN; ZHU, 2015) e no *Gigawords* inglês (NAPOLES; GORMLEY; VAN DURME, 2012), demonstraram que o modelo proposto superou os modelos referências, sendo que a análise mostrou que o modelo foi capaz de gerar resumo de maior qualidade e reduzir repetições.

(SONG; ZHAO; LIU, 2018) apresentaram mecanismos de cópia com estrutura de infusão, para facilitar a cópia de palavras e relações importantes da frase de origem com a de resumo. Para isso, apresentaram uma nova arquitetura neural que ajuda na preservação de palavras ou relações fundamentais no resumo. O *framework* combina a estrutura da árvore de

análise de dependência com o mecanismo de cópia de um sistema de resumo abstrativo. Os experimentos mostraram a eficácia em incorporar informações sintáticas na fonte do sistema, sendo a abordagem proposta comparada de forma promissora aos métodos do estado da arte.

(LI; BING; LAM, 2018) apresentaramm um *framework* para treinamento e sumarização neural abstrativa baseado em uma abordagem "ator-crítico"de aprendizado por reforço. Para o "ator", empregaram o *framework sequence-to-sequence* como a rede política para geração de resumo. Para o "crítico", combinaram a estimativa por máxima verossimilhança com um bem projetado estimador global de qualidade de resumo, sendo um classificador binário baseado em rede neural. O método de gradiente de política - busca aprender uma política parametrizada, aumentando ou diminuindo a probabilidade de suas ações com base na recompensa recebida em cada estado - é utilizado para conduzir o aprendizado dos parâmetros. Experimentos extensos em conjuntos de dados *benchmark*, em diferentes linguagens, mostraram que o *framework* proposto alcançou melhores resultados em relação aos métodos do estado da arte.

(SONG; HUANG; RUAN, 2019) propuseram um *framework* abstrativo de sumarização de texto nomeado ATSDL baseado em LSTM-CNN que constrói novas sentenças explorando mais fragmentos refinados do que frases, ou seja, frases semânticas. Para alcançar esse objetivo, os autores aplicaram o modelo *Long Short-Term Memory* (LSTM) (HOCHREITER; SCHMIDHUBER, 1997) - que foi desenvolvido originalmente para tradução automática - para sumarização, e combinaram uma Rede Neural Convolutiva (CNN) e LSTM para melhorar o desempenho da sumarização de texto. ASTDL considera estruturas semânticas e sintáticas, o que é diferente dos modelos existentes de sumarização extrativa e abstrativa. ASTDL primeiro aplica o método de extração de frase para extrair frases-chave das sentenças e, na sequência, utiliza o modelo LSTM para aprender a colocação nas frases. Depois de treinar, o novo modelo gera a sequência das frases. Essa sequência é o texto sumarizado composto de sentenças naturais. Resultados de experimentos no conjunto de dados *CNN/Daily Mail* mostraram que o *framework* ASTDL superou os modelos no estado da arte em termos de estrutura semântica e sintática, além disso, atingiu resultados competitivos na avaliação da qualidade linguística manual.

(RUSH; CHOPRA; WESTON, 2015) propuseram uma abordagem totalmente orientada a dados para sumarização abstrativa de texto. Os autores utilizaram um modelo local baseado em atenção que gera cada palavra do resumo por meio de condições na sentença de entrada. Enquanto o modelo é estruturalmente simples, pode ser facilmente treinado *end-to-end* e escalo-

nado para grande volume de dados de treinamento. O modelo apresentou ganhos significativos de desempenho e apresentou resumos abstrativos precisos.

(CHOPRA; AULI; RUSH, 2016) propuseram uma nova rede neural recorrente para o problema de sumarização abstrativa de sentença. O método proposto consiste em uma rede neural recorrente condicional, que atua como um decodificador para gerar o resumo de uma sentença de entrada, muito semelhante a um modelo de linguagem padrão. Além disso, a cada passo, o decodificador também recebe uma entrada de condicionamento que é a saída de um módulo codificador. Os experimentos mostraram que o modelo superou significativamente os métodos recentes até 2016 propostos no estado da arte no conjunto de dados *Gigawords*.

(TAN; WAN; XIAO, 2017) propuseram um novo mecanismo de atenção baseado em grafo em um *framework sequence-to-sequence* para sumarização abstrativa de textos. Os autores investigaram os desafios de gerar longas sequências para modelos *sequence-to-sequence*, e propuseram um novo algoritmo de decodificação hierárquica com um mecanismo de referência para gerar os resumos abstratos. Resultados experimentais mostraram que o modelo proposto é capaz de obter melhoria considerável em relação aos modelos de sumarização abstrativa neurais anteriores.

(CAO et al., 2018) propuseram o uso de resumos existentes como modelos flexíveis para orientar o modelo *sequence-to-sequence*. Para isso, utilizaram uma plataforma de recuperação de informação popular (Apache Lucene) para recuperar resumos adequados como candidatos padrões. Em seguida, estenderam a estrutura *sequence-to-sequence* para conduzir conjuntamente a re-classificação e geração de resumo com reconhecimento de padrão. Experimentos mostraram que o modelo proposto pode gerar resumos informativos, legíveis e estáveis.

(CELIKYILMAZ et al., 2018) apresentaram agentes de comunicação profunda - divide a tarefa de codificar um texto longo com múltiplos agentes codificadores colaborativos - em uma arquitetura *encoder-decoder* para enfrentar os desafios de representar um longo documento para resumos abstrativos. Utilizando agentes de comunicação profunda, a tarefa de codificar um texto longo é dividida em vários agentes colaboradores, cada um reponsável por uma subseção do texto de entrada. Esses codificadores são conectados a um único decodificador, treinado de ponta a ponta usando o aprendizado por reforço para gerar um resumo focado e coerente. A análise executada demonstrou melhoria devido à capacidade aprimorada de cobrir somente os conceitos essenciais e manter a coerência semântica nos resumos.

(GEHRMANN; DENG; RUSH, 2018) propuseram um método de seleção de conteúdo para sumarização abstrativa. Para isso, utilizaram atenção *bottom-up* para sumarização abstra-

tiva neural. A abordagem dos autores primeiro seleciona uma máscara de seleção - ressalta as partes que devem ser extraídas do texto - para o documento fonte e então restringe um modelo neural por esta máscara. Essa abordagem pode ter uma maior precisão em decidir quais frases um modelo deve incluir numa sumarização, sem sacrificar as vantagens da fluência da sumarização abstrativa. Resultados dos experimentos mostraram que o sistema de sumarização combinado com *bottom-up* levou a melhorias na métrica ROUGE em mais de dois pontos nos conjuntos de dados *CNN/Daily Mail* (HERMANN et al., 2015) e NYT (SANDHAUS, 2008).

(GUO; PASUNURU; BANSAL, 2018) aprimoraram a sumarização de texto abstrativa por meio de um aprendizado específico de múltiplas tarefas por meio de aprendizado multitarefa com tarefas auxiliares de geração de perguntas e geração de consequências. Para isso, utilizaram a tarefa de geração de perguntas, para ensinar o modelo de sumarização como procurar detalhes importantes dignos de questionamento. Além disso, utilizaram a tarefa de geração de vínculos para ensinar o modelo a reescrever um resumo como um subconjunto lógico direcionado do documento de entrada. Assim, obtiveram melhorias estatiscamente siginificativas em relação ao estado da arte nos conjuntos de dados *CNN/Daily Mail* e *Gigawords*, assim como no DUC-2002.

(KRYŚCIŃSKI et al., 2018) propuseram duas técnicas para melhorar o nível de abstração dos resumos gerados. Na primeira, decomporam o decodificador em uma rede de contexto que recupera partes relevantes do documento de origem. Na segunda, utilizaram um modelo de linguagem pré-treinado que incorpora conhecimento prévio sobre geração de linguagem. O trabalho mostrou que o modelo proposto gerou resumos muito mais abstrativos do que as abordagens anteriores, mantendo altas pontuações na métrica utilizada (ROUGE) próximas ou acima do estado da arte.

(WANG et al., 2018) propuseram uma abordagem de *deep learning* para as tarefas de sumarização automática, incorporando tópicos de informações no modelo convolucional *sequence-to-sequence* além de utilizar treinamento sequencial autocrítico para otimização. Os resultados empíricos demonstraram superioridade do método proposto na sumarização abstrativa.

2.3 MODELOS HÍBRIDOS DE SUMARIZAÇÃO EXTRATIVA E ABSTRATIVA

(RUDRA et al., 2016) propuseram um *framework* que primeiro atribui *tweets* a diferentes classes de situações e depois resume de forma automática esses *tweets*. O *framework*

proposto possui dois estágios de resumo. O primeiro extrai um conjunto de *tweets* importantes de todo o conjunto de informações - por meio de uma técnica de otimização baseada em programação linear inteira. O segundo estágio utiliza um grafo de palavras e uma técnica de sumarização abstrativa baseada em conteúdo de palavras para produzir a sumarização final. Experimentos mostraram que o modelo proposto performou de forma superior ao *baseline*.

(ZENG et al., 2016) propuseram um mecanismo de cópia simples capaz de explorar vocabulários pequenos e manipular palavras fora do vocabulário. Para isso, introduziram uma metodologia que lê, inicialmente, a sequência de entrada antes de se comprometer com a representação de cada palavra. A primeira representação de leitura influencia a segunda representação e, portanto, permite que os vetores ocultos intermediários capturem o significado apropriado para o texto de entrada. Experimentos realizados nos conjuntos de dados *Gigawords* e DUC mostraram a eficácia da abordagem proposta, superando o estado da arte.

2.4 TRABALHOS DE REVISÃO SOBRE SUMARIZAÇÃO DE TEXTO

(KEDZIE; MCKEOWN; DAUME III, 2018) buscaram compreender como os modelos de resumo baseados em aprendizagem profunda executam a seleção de conteúdo em diversos domínios: notícias, histórias pessoais, reuniões e artigos médicos. Os autores analisaram diferentes arquiteturas de redes neurais para extração de sentenças. Sendo assim, compararam as representações de sentenças baseadas em redes neurais recorrentes e redes neurais convolutivas com a abordagem mais simples de *word embedding* para entender os ganhos de arquiteturas mais sofisticadas. Assim, mostraram experimentalmente que os modelos sofisticados, que utilizam aprendizado profundo, de resumo não melhoram o desempenho quando comparados aos modelos mais simples, que não utilizam aprendizado profundo.

(SHI et al., 2018) apresentaram uma literatura abrangente e uma pesquisa técnica sobre diferentes modelos *sequence-to-sequence*, para sumarização abstrativa de texto, incluindo estruturas de rede, estratégias de treinamento e geração de sequência. Para isso, mostraram diferentes perspectivas nos avanços dos modelos *sequence-to-sequence*. Por fim, resumiram os resultados dos experimentos de diferentes modelos *sequence-to-sequence* para o conjunto de dados *CNN/-Daily Mail*, mostrando que os trabalhos de (KRYŚCIŃSKI et al., 2018), (CELIKYILMAZ et al., 2018) e (GEHRMANN; DENG; RUSH, 2018) obtiveram melhores resultados.

2.5 MODELOS COMPUTACIONAIS COGNITIVOS

Nesta seção, são apresentados os modelos computacionais cognitivos: CLARION (SUN, 1997), IDA (FRANKLIN; KELEMEN; MCCAULEY, 1998), LIDA (FRANKLIN; PATTERSON JR, 2006), CERA (MORENO; ESPINO; DE MIGUEL, 2007), CODAM (TAYLOR, 2007), para o conhecimento de como são desenvolvidos esses trabalhos. Os artigos apresentados são, principalmente, do modelo LIDA, por ser o escolhido para a execução deste trabalho. Com o objetivo de demonstrar a evolução histórica do assunto, os trabalhos são apresentados em ordem cronológica.

O LIDA, como poderá ser visto abaixo, de forma sucinta, é o modelo de evolução do IDA que apresenta módulos teóricos e computacionalmente desenvolvidos, os quais podem ser utilizados para aperfeiçoar partes dos processos da tarefa de sumarização automática de texto na abordagem abstrativa.

Em (SUN, 1997) foi examinada a aprendizagem explícita e implícita em diferentes experimentos psicológicos, esboçando a distinção consciente/inconsciente olhando para dois tipos de aprendizagem. Para isso, desenvolveu o modelo CLARION (*Connectionist Learning with Adaptative Rule Induction On-line*), no qual foi aplicado Redes Neurais e técnicas de Aprendizado de Máquina para debater o complexo aprendizado humano e a consciência humana. O artigo não apresenta resultados concretos, mas avalia que em futuros trabalhos será necessário trabalhar com um alto nível de complexidade.

(FRANKLIN; KELEMEN; MCCAULEY, 1998) descreveram uma arquitetura para um agente de distribuição inteligente projetado para a marinha. Esse agente de software autônomo implementa o Global Workspace, uma teoria psicológica da consciência, que possibilita que o agente consiga lidar com novos problemas. Com o objetivo de mostrar o progresso do projeto, é exposto como resultado que o IDA é uma prova do conceito de software consciente.

(FRANKLIN, 2000) descreveram as arquiteturas de dois agentes de software "conscientes", além dos modelos conceituais e computacionais relativamente abrangentes derivados deles. Sendo assim, os autores utilizam a teoria de Baars do Global Workspace, a arquitetura de seleção de alta ação de Sloman, os conceitos de Glenberg como teoria de ações, os sistemas de símbolos perceptivos de Barsalou e a hipótese dinâmica. Embora não apresentem resultados conclusivos, sugerem que o estudo ajude na utilização de agentes de software para modelagem cognitiva.

(JOHNSON; CAULFIELD; TAYLOR, 2000) apresentam e aplicam um sistema de controle de *feedback* unidimensional. Trata-se de um algoritmo de consciência que usa simulação e avaliação preditivas para permitir que o sistema de exemplo reaprenda novos modelos internos e externos. Por fim, apresentaram como discussão o pioneirismo do trabalho como o primeiro sistema consciente artificial o ARTCON 1.

(FRANKLIN, 2003) descreveram o IDA (*Intelligent Distributed Agent*), um agente de software autônomo e "consciente"mostrando como ele interage com seres humanos. O objetivo do IDA é conversar com os marinheiros da marinha dos EUA utilizando linguagem natural para chegar a uma situação de trabalho que ao mesmo tempo seja favorável para o marinheiro e para a marinha. O IDA foi utilizado pelos marinheiros os quais reportaram que as tarefas foram realizadas como eles as fariam.

(FRANKLIN; PATTERSON JR, 2006) apresentaram a arquitetura LIDA (*Learning Intelligent Distributed Agent*) baseado no IDA. Para isso, foram acrescentados três módulos à arquitetura do IDA: aprendizado perceptivo, aprendizado episódico e aprendizado processual. Sendo assim, o LIDA foi projetado para aprender com a experiência, o que pode gerar várias lições ao longo de vários ciclos cognitivos. A arquitetura é útil para gerar conjecturas testáveis sobre como a mente humana funciona, utilizando uma abordagem sistêmica da cognição.

(MORENO; ESPINO; DE MIGUEL, 2007) descreveram a aplicação de um novo modelo – CERA (*Conscious and Emotional Reasoning Architecture*) - de consciência de máquina a um problema específico de exploração de ambiente desconhecido. Para isso, analisam o que as capacidades cognitivas, como atenção, consciência ambiental e aprendizado emocional, podem oferecer. O modelo desenvolvido integra esses conceitos em um framework agente de controle localizado, sendo sua primeira versão testada em um simulador de robótica avançado. O CERA apresentou bons resultados na identificação e associação de ambientes, porém, mostrou dificuldades na representação de espaços tridimensionais.

(TAYLOR, 2007) apresentaram uma revisão do modelo de consciência CODAM (*Corollary Discharge of Attention Movement*) e o desenvolveram para chegar a um modelo funcional da consciência. Sendo assim, o desenvolvimento foi orientado pela análise da natureza da consciência, utilizando o modelo CODAM para gerar um conjunto de previsões sobre a velocidade de resposta de sistemas de controle. O modelo apresentou eficiência em alguns experimentos, porém necessita de melhorias em muitos pontos, o que pode gerar modificações futuras no modelo.

(VASSEV; HINCHEY, 2013) apresentaram a implementação da consciência artificial por meio do *framework KnowLang*. Para isso, o *framework* fornece um contexto de conhecimento especial e um raciocínio especial que opera nesse contexto. Sendo assim, o raciocínio se comunica com o sistema principal por meio de operadores especiais de solicitação e entrega, permitindo conclusões e atualizações de consciência. Os resultados apresentados mostraram que se aproximam do comportamento humano com base na atenção aplicada.

(FRANKLIN et al., 2013) descreveram a arquitetura LIDA (*Learning Intelligent Distributed Agent*), a qual combina seleção sofisticada de ações, motivação, um importante mecanismo central de atenção e aprendizado multimodal por instrução e seleção. Por fim, indicaram que o LIDA pode ser utilizado tanto para simulações quanto para o mundo real.

(FRANKLIN et al., 2016) apresentaram um tutorial com descrição detalhada e completa do modelo conceitual LIDA. Primeiro expuseram os conceitos do LIDA e na sequência, introduziramm o modelo computacional do LIDA, por meio do *framework* LIDA. Além disso mostraram esboços de alguns agentes desenvolvidos baseados no LIDA. O modelo possui módulos que são conceitualmente estabelecidos, porém, não estão implementados no *framework*.

(AGRAWAL; FRANKLIN; SNAIDER, 2018) propuseram um novo sistema de memória sensorial para o LIDA que complementa o novo sistema representacional baseado em vetor do Vetor LIDA - foi apresentado como uma grande revisão do sistema de representação do modelo LIDA - fornecendo representações mais ricas do que nós e links. As representações sensoriais são na forma de representações distribuídas esparsas de alta dimensão. Além disso, descrevem um método de conversão de alta fidelidade entre representação distribuída esparsa e vetores de representação composta modular - vetores de inteiros longos - que preservam a riqueza de informações durante a conversão. Concluem que o sistema é capaz de reconstruir as representações esparsas com bons números de recall e precisão.

(KUGELE; FRANKLIN, 2020a) introduzem um dicionário de conceitos relacionados à ativação e uma taxonomia que enumera muitos fatores padrões relacionados à ativação que tenham aparecido em arquiteturas cognitivas. Sendo assim, no artigo os autores demonstram a taxonomia aplicando-a à arquitetura cognitiva do LIDA, que inclui uma das mais variadas e abrangentes aplicações da funcionalidade relacionada à ativação. Os autores sugerem que alguns testes devam ser aplicados para confirmar a validade e utilidade da taxonomia.

(KUGELE; FRANKLIN, 2020b) apresentam uma revisão detalhada do aprendizado no LIDA, sendo assim elaboram e sintetizam ideias de diferentes trabalhos, usando terminologia atualizada que reflete a evolução do LIDA. Para isso, exploram questões fundamentais na apren-

dizagem, como, "O que de ser deve ser contruído diretamente no LIDA?"contra "O que deve ser aprendido pelo LIDA?", a natureza das representações do LIDA e a relação entre o modelo conceitual do LIDA e suas realizações computacionais. Por fim, os autores acreditam que este artigo direciona os esforços de pesquisa e fornece uma introdução completa aos fundamentos conceituais dos mecanismos de aprendizado do LIDA, além disso afirmam que o artigo pode ser útil para qualquer pessoa que deseja um entendimento mais profundo do LIDA ou para aqueles que planejam implementar agentes baseados no LIDA.

3 CONCEITOS FUNDAMENTAIS

Neste capítulo serão apresentados os principais conceitos utilizados neste trabalho, incluindo as técnicas de sumarização de texto e as definições teóricas do modelo cognitivo LIDA.

3.1 SUMARIZAÇÃO ABSTRATIVA

Sumarização abstrativa é a tarefa de gerar resumo utilizando novas palavras e frases não incluídas no texto de origem (Figura 3), como costuma acontecer em resumos feitos por humanos (SEE; LIU; MANNING, 2017). É utilizado o adjetivo "abstrativo" para denotar a sumarização que não é uma mera seleção de sentenças já existentes ou extraídas de um texto, mas uma paráfrase comprimida do conteúdo principal do mesmo, possivelmente utilizando vocabulário não visto no texto origem (NALLAPATI et al., 2016).

O sucesso no desenvolvimento de modelos de sumarização abstrativa se deu, majoritariamente, com o surgimento dos modelos *sequence-to-sequence* (seq2seq) (SUTSKEVER; VINYALS; LE, 2014), os quais utilizam principalmente redes neurais recorrentes (RNNs), *long short-term memory* (LSTM) e o mecanismo de atenção de (BAHDANAU; CHO; BENGIO, 2014). Os modelos abstrativos em documentos longos sofrem lentidão e são imprecisos, sendo necessário o uso do mecanismo de atenção. Além disso, sofrem de redundância (repetição) especialmente na geração de resumo em múltiplas sentenças (CHEN; BANSAL, 2018), por isso em alguns trabalhos são utilizadas *Pointer Networks*. Essas e outras irregularidades são estudadas atualmente na literatura, a fim de aperfeiçoar ainda mais a geração desse tipo de resumo.

Por fim, algumas pesquisas mostram que os resumos escritos por humanos são mais abstratos, como pode ser visto em (SEE; LIU; MANNING, 2017), os quais revelam que as pessoas podem seguir naturalmente algumas estruturas inerentes quando escrevem resumos abstratos.

Figura 3 – Exemplo de resumo extrativo e abstrativo. Nota-se que no resumo extrativo foram extraídos trechos do texto original, destacados em itálico e negrito, e em contrapartida no resumo abstrativo é possível observar que foram geradas novas palavras para construir o resumo.

<table>
<tr><th>Texto Original</th><th>Resumo Extrativo</th></tr>
<tr><td rowspan="3">O governo de São Paulo colocou todo o estado na fase amarela do plano de flexibilização econômica. O anúncio foi feito durante coletiva de imprensa no início da tarde desta segunda-feira (30). O estado registra 42.095 mortes por Covid-19 e 1,24 milhão de casos confirmados da doença desde o início da pandemia. A fase amarela no plano de flexibilização é mais restritiva que a verde e limita mais os horários de funcionamento do comércio e serviços, por exemplo. Seis regiões, entre elas a capital paulista, regridem da fase verde para a amarela. As demais 11 regiões, que já estavam na fase amarela, não avançam e seguem no mesmo estágio. A previsão é a de que o decreto que coloca as regiões na fase amarela seja publicado na terça-feira (1) e a medida comece a valer na quarta-feira (2). Além disso, o governo mudou novamente os critérios para a reclassificação. Se tivessem sido mantidos parâmetros anteriores, a Grande São Paulo, que incluiu a capital, teria piora suficiente para migrar para a fase laranja, ainda mais restritiva.
"Com o claro aumento da instabilidade da pandemia, o governo do estado de São Paulo e o Centro de Contingência da Covid-19 decidiram que 100% do estado vai retornar para a fase amarela do Plano SP. Essa medida, quero deixar claro, não fecha comércio, nem bares, nem restaurantes. A fase amarela não fecha atividades econômicas, mas é mais restritiva nas medidas para evitar aglomerações e o aumento do contágio", disse o governador João Doria (PSDB). A atualização da reclassificação foi divulgada menos de 24 horas após as eleições municipais e só foi permitida por conta de novas alterações feitas nas regras do plano. No dia 13 de novembro, Doria gravou um vídeo dizendo que o endurecimento das medidas de combate à pandemia após as eleições eram fake news.
"Meu repúdio a mais uma fake news. Não vamos fechar o comércio ou endurecer as medidas de combate à pandemia após as eleições. Mais um absurdo que estão inventando", disse em sua conta no Twitter.</td><td>O governo de São Paulo colocou todo o estado na fase amarela do plano de flexibilização econômica. A fase amarela não fecha atividades econômicas, mas é mais restritiva nas medidas para evitar aglomerações e o aumento do contágio ", disse o governador João Doria (PSDB). No dia 13 de novembro, Doria gravou um vídeo dizendo que o endurecimento das medidas de combate à pandemia após as eleições eram fake news. Não vamos fechar o comércio ou endurecer as medidas de combate à pandemia após as eleições.</td></tr>
<tr><th>Resumo Abstrativo</th></tr>
<tr><td>Menos de 24 horas após o fim do 2º turno das eleições municipais, o governo de São Paulo deu um passo atrás no plano de reabertura e colocou todo o estado na fase amarela — que limita mais os horários de funcionamento do comércio e serviços, por exemplo. O estado registra 42.095 mortes por Covid e 1,24 milhão de casos confirmados desde o início da pandemia.</td></tr>
</table>

Fonte: adaptado de G1; MACHADO et al., 2020; 2020

3.2 REDES NEURAIS RECORRENTES

As redes neurais recorrentes (RNN) (RUMELHART; HINTON; WILLIAMS, 1986) são uma família de redes neurais para processamento de dados sequenciais que são utilizadas para tarefas como: tradução, sumarização de texto e classificação de sentimento, entre outras. Diferentemente de uma rede neural *perceptron*, as redes neurais recorrentes compartilham os mesmos pesos ao longo do tempo. Para isso, cada membro da saída é produzido utilizando a mesma regra de atualização aplicada às saídas anteriores. Sendo assim, o compartilhamento de parâmetros ocorre através de uma rede profunda com diversos nós (GOODFELLOW; BENGIO; COURVILLE, 2016).

Diferentes redes neurais recorrentes utilizam a Equação (1) - sendo, h a representação do estado e x o vetor de entrada - ou semelhante para definir os valores das unidades da camada oculta, a qual demonstra a atualização das informações dos estados anteriores $h^{(t-1)}$ ao atual $h^{(t)}$, fazendo com que os parâmetros sejam compartilhados de forma recorrente (GOODFELLOW; BENGIO; COURVILLE, 2016).

$$h^{(t)} = f(h^{(t-1)}, x^{(t)}) \tag{1}$$

Na Figura 4, é ilustrada uma rede recorrente que produz uma saída para cada etapa do tempo t e possui conexões recorrentes entre as unidades da camada oculta. É assumido que a rede possui função de ativação da tangente hiperbólica, assim como é assumido que a saída é discreta. Sendo assim, é aplicada a operação *softmax* como uma etapa de pós-processamento para obter um vetor $\hat{y}$ de probabilidades normalizadas da saída. A propagação direta inicia com uma especificação do estado inicial $h^{(0)}$. Assim, para cada passo de $t = 1$ a $t = \tau$, sendo τ o número de entradas, é aplicado o seguinte conjunto de equações (GOODFELLOW; BENGIO; COURVILLE, 2016):

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)} \tag{2}$$

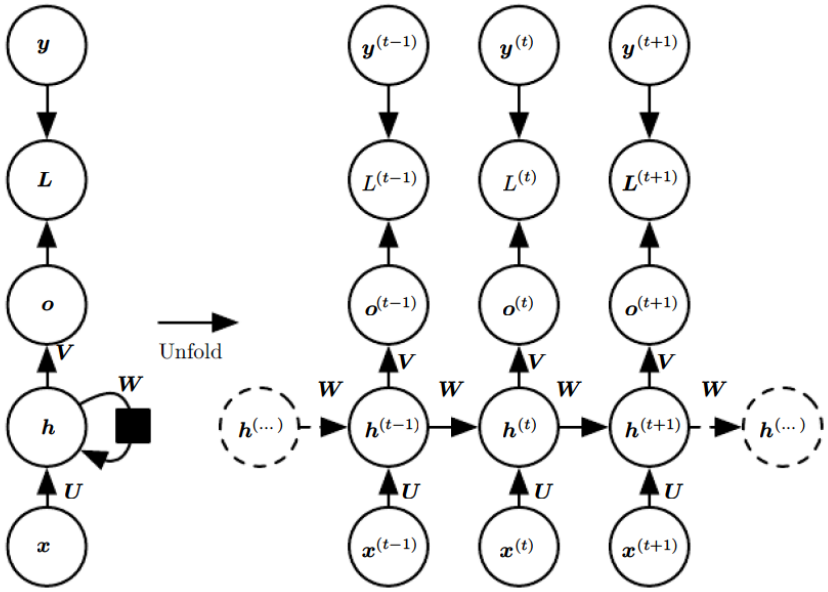
$$h^{(t)} = tanh(a^{(t)}) \tag{3}$$

$$o^{(t)} = c + Vh^{(t)} \tag{4}$$

$$\hat{y}^{(t)} = softmax(o^{(t)}) \tag{5}$$

onde os parâmetros b e c são os vetores de viés, sendo as matrizes de peso U, V e W, respectivamente, da entrada para a camada oculta, da camada h para a saída o e das conexões das unidades da camada oculta (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 4 – Representação de uma RNN. À esquerda o diagrama de circuito e à direita a representação gráfica extendida. Em cada etapa do tempo t, as entradas são $x^{(t)}$, as ativações da camada oculta são $h^{(t)}$, as saídas são $o^{(t)}$, as marcações são $y^{(t)}$ e a perda é $L^{(t)}$

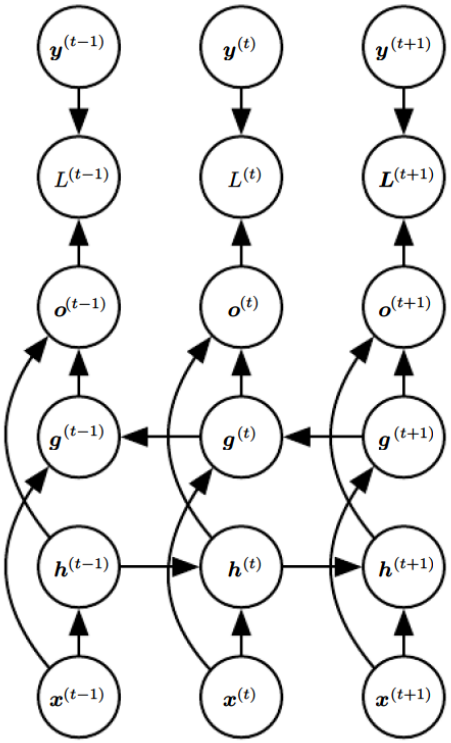


Fonte: GOODFELLOW; BENGIO; COURVILLE, 2016

3.2.1 Redes Neurais Recorrentes Bidirecionais

Uma rede neural recorrente bidirecional (BRNN) combina uma RNN que se move para frente através do tempo iniciando do começo de uma sequência, com uma RNN que se move para trás através do tempo iniciando do fim de uma sequência. A Figura 5 ilustra uma típica BRNN, com $h^{(t)}$ representando a sub-RNN que se move para frente através do tempo e $g^{(t)}$ representando a sub-RNN que se move para trás através do tempo. Isso permite que as unidades de saída $o^{(t)}$ computem a representação que depende tanto do passado quanto do futuro (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 5 – Representação de uma BRNN. Em cada etapa do tempo t, as entradas são $x^{(t)}$, as ativações da camada oculta que propagam informação para frente através do tempo são $h^{(t)}$, as ativações da camada oculta que propaga informação para trás através do tempo são $g^{(t)}$ as saídas são $o^{(t)}$, as marcações são $y^{(t)}$ e a perda é $L^{(t)}$.



Fonte: GOODFELLOW; BENGIO; COURVILLE, 2016

3.2.2 Memória de curto-longo prazo

A memória de curto-longo prazo ou *long short-term memory* (LSTM) proposta por (HOCHREITER; SCHMIDHUBER, 1997) é uma arquitetura de RNN construída para lidar com sequências temporais e capturar dependências de longo prazo de forma mais aprimorada do que as RNNs convencionais.

O problema das RNNs convencionais, como a mostrada na Seção 3.2, é que, durante o treinamento, os componentes do vetor gradiente podem crescer ou decrescer exponencialmente ao longo de algumas sequências. Esse problema com explosão ou perda de gradiente, também conhecido, respectivamente, como *exploding gradient* e *vanishing gradient*, dificulta o aprendizado da RNN em correlações de longa distância numa sequência (TAI; SOCHER; MANNING, 2015). Sendo assim, a LSTM resolve esse problema introduzindo uma célula de memória capaz de preservar o estado por longos períodos no tempo.

A LSTM contém unidades especiais chamadas de blocos de memória na camada oculta recorrente. Os blocos de memória, como podem ser visto na Figura 6, contêm células de memória com conexões entre si que guardam o estado temporal da rede, além de unidades multiplicativas especiais chamadas de portas para controlar o fluxo de informação (SAK; SENIOR; BEAUFAYS, 2014). Dentro dos blocos de memória existe a porta de esquecimento, a porta de entrada e a porta de saída. A porta de esquecimento é responsável por definir o que deve ser preservado a cada instante na célula de memória; a porta de entrada controla o fluxo das ativações de entrada dentro da célula de memória; a porta de saída controla o fluxo das ativações dentro do restante da rede (SAK; SENIOR; BEAUFAYS, 2014).

Diferentes arquiteturas de LSTM foram desenvolvidas, sendo assim, a versão abaixo é baseada na descrita por (TAI; SOCHER; MANNING, 2015). As unidades da LSTM são determinadas em cada etapa do tempo t como um conjunto de vetores em $\mathbb{R}^d$: uma porta de esquecimento f_t, uma porta de entrada i_t, uma porta de saída o_t, uma célula de memória c_t e um estado da camada oculta ou bloco de memória h_t, onde $i_t, f_t, o_t \in [0:1]$. A dimensão da memória da LSTM é definida como d.

As equações de transição da LSTM são definidas nas expressões de 6-11.:

$$f_t = \sigma(W^{(f)}x_t + U^{(f)}h_{t-1} + b^{(f)}) \tag{6}$$

$$i_t = \sigma(W^{(i)}x_t + U^{(i)}h_{t-1} + b^{(i)}) \tag{7}$$

$$o_t = \sigma(W^{(o)}x_t + U^{(o)}h_{t-1} + b^{(o)}) \tag{8}$$

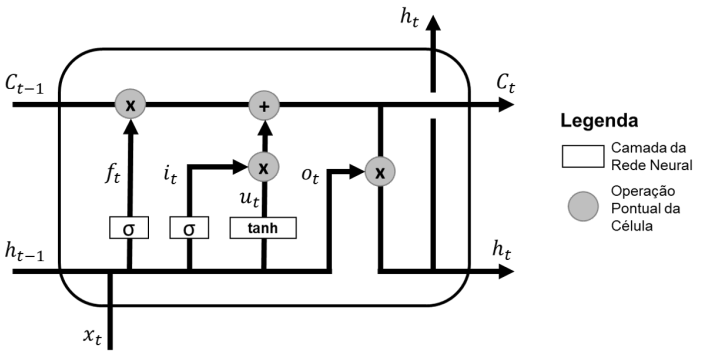
$$u_t = tanh(W^{(u)}x_t + U^{(u)}h_{t-1} + b^{(u)}) \tag{9}$$

$$c_t = i_t \odot u_t + f_t \odot c_{t-1} \tag{10}$$

$$h_t = o_t \odot c_t \tag{11}$$

onde x_t é a entrada na atual etapa do tempo, σ é a função *sigmoid* logística e $\odot$ é a unidade multiplicativa.

Figura 6 – Representação de um bloco de memória da LSTM



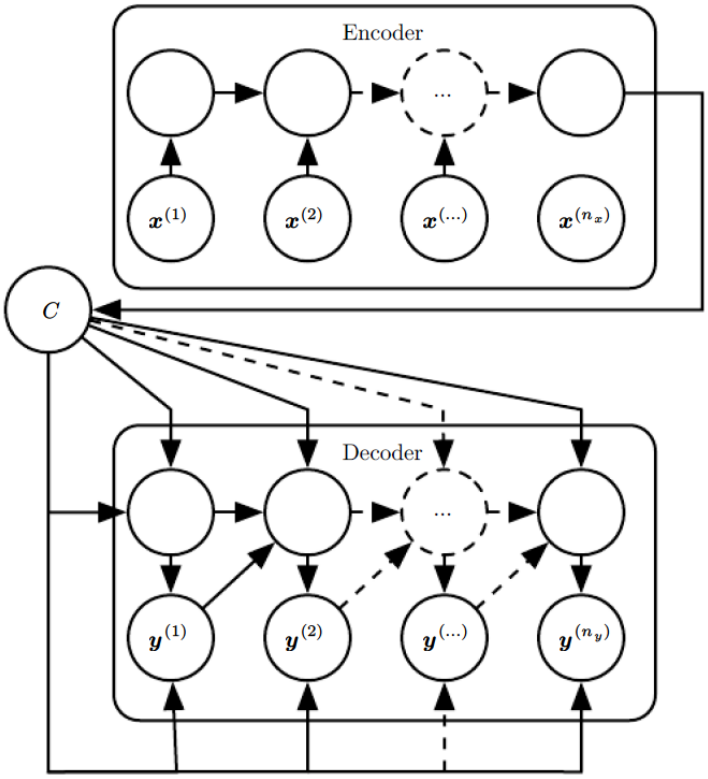
Fonte: adaptado de OLAH, 2015

3.2.3 Arquitetura *Sequence-to-Sequence*

A aplicação e implementação, com sucesso, de uma arquitetura *sequence-to-sequence* (seq2seq) foi proposta por (SUTSKEVER; VINYALS; LE, 2014) para a tarefa de tradução (GOODFELLOW; BENGIO; COURVILLE, 2016), em que a arquitetura seq2seq é aplicada quando as sequências de entrada e saída são de tamanhos diferentes. Isso ocorre, por exemplo, na tarefa de sumarização de texto, na qual é dado um texto como entrada (sequência de entrada) sendo esperado como saída o resumo desse texto (sequência de saída); ou seja, a redução da sequência de entrada.

Também conhecida como arquitetura *encoder-decoder*, a seq2seq é composta de um codificador que recebe os dados da sequência de entrada e de um decodificador que gera a sequência de saída. Ambas as partes (codificador e decodificador) são como dois modelos distintos combinados dentro de um maior, como pode ser visto na Figura 7. O codificador converte a sequência de entrada em um vetor da camada oculta, o qual representa seu contexto. O decodificador recebe como entrada o vetor de contexto gerado pelo codificador para produzir o próximo vetor da camada oculta para enfim gerar a sequência de saída. Sendo assim, essa arquitetura mostra-se inovadora (GOODFELLOW; BENGIO; COURVILLE, 2016) por conseguir receber como entrada uma sequência de um determinado tamanho e gerar em sua saída uma sequência de outro tamanho.

Figura 7 – Exemplo de uma arquitetura Seq2Seq onde as sequências de entrada e saída são representadas, respectivamente, por $(x^{(1)}, x^{(2)},..., x^{(n_x)})$ e $(y^{(1)}, y^{(2)},..., y^{(n_y)})$ nela, o vetor de contexto é representado por C. É possível verificar no codificador que a sequência de entrada é convertida em um vetor de contexto através da camada oculta. O decodificador então recebe esse vetor de contexto e o transforma na sequência de saída.



Fonte: GOODFELLOW; BENGIO; COURVILLE, 2016

3.2.4 Mecanismo de Atenção

O Mecanismo de Atenção proposto em (BAHDANAU; CHO; BENGIO, 2014) surgiu para ajudar a memorizar longas sentenças para a tarefa de tradução automática. A maioria dos modelos para essa tarefa, até o momento que foi proposto o Mecanismo de Atenção, utilizavam *encoder-decoders*. Uma rede neural codificadora lê e codifica uma sentença em um vetor de tamanho fixo. Um decodificador então gera uma tradução do vetor codificado. Todo o sistema de codificação e decodificação é treinado em conjunto para maximizar a probabilidade da tradução correta de uma sentença.

Um problema com a abordagem *encoder-decoder* é que a rede neural deve ser capaz de comprimir toda informação necessária da sentença de entrada em um vetor de tamanho fixo,

tornando a tarefa de copiar longas sentenças difícil, especialmente aquelas que são maiores que as sentenças de treinamento.

Para resolver esse problema, em (BAHDANAU; CHO; BENGIO, 2014) foi proposto o mecanismo de atenção, uma extensão do modelo *encoder-decoder*. A cada momento, o modelo gera uma palavra numa tradução, procurando então por um conjunto de posições da sentença onde podem estar concentradas as informações mais relevantes. Assim, o modelo prevê a palavra alvo baseado nos vetores de contexto associados às posições e em todas as outras previsões das palavras alvo.

Essa abordagem permite que o modelo não codifique toda a sentença de entrada em um único vetor de tamanho fixo. Sendo assim, o modelo codifica as sentenças de entrada numa sequência de vetores e seleciona um subconjunto desses vetores de forma adaptada enquanto decodifica a tradução. Isso libera o modelo de tradução automática de ter que comprimir toda a informação da sentença - independentemente de seu tamanho - em um vetor de tamanho único.

Em (SHI et al., 2018) é demonstrado o uso do mecanismo de atenção para a tarefa de sumarização abstrativa, como pode ser observado na Figura 8 numa arquitetura Seq2Seq baseada em atenção. Nela, o decodificador não utiliza apenas as representações do texto de entrada vindas do codificador (camada oculta final e células de estado), mas também foca de forma seletiva em partes do artigo fonte em cada passo da decodificação.

Para explicar melhor o funcionamento da arquitetura, em (SHI et al., 2018) é dado um exemplo de sumarização no qual se supõe o resumo da seguinte frase: “Kylian Mbappe marcou dois gols aos quatro minutos do segundo tempo para classificar a França às quartas de final da Copa do Mundo num emocionante 4-3 na vitória contra a Argentina no Sábado”, tendo sua versão resumida como: “França bate a Argentina por 4-3 e vai às quartas de final”. Quando a palavra “bate” é gerada, o decodificador precisa dar maior atenção ao trecho “emocionante 4-3 na vitória” do que aos outros trechos do texto. Essa atenção é feita por meio do uso do mecanismo de atenção proposto por Bahdanau, o qual, primeiro calcula a distribuição de atenção das palavras do texto fonte, permitindo na sequência que o decodificador saiba onde dar atenção para produzir a palavra/frase do resumo alvo. Na arquitetura Seq2Seq da Figura 8, dado todos os estados da camada oculta do codificador $h^e = (h_1^e, h_2^e, ..., h_j^e)$ e o estado atual da camada oculta do decodificador h_t^d, a distribuição de atenção α_t^e sobre os *tokens* do codificador é calculada de acordo com a Expressão (12) a seguir. :

$$\alpha_{tj}^e = \frac{exp(s_{tj}^e)}{\sum_{k=1}^{J} exp(s_{tk}^e)} \tag{12}$$

onde a pontuação de atenção $s_{tj}^e = s(h_j^e, h_t^d)$ é obtida pela função de pontuação de conteúdo base, que possui três alternativas, expressadas nas Equações 13-15, como sugerido em (LUONG; PHAM; MANNING, 2015):

$$s(h_j^e, h_t^d) = (h_j^e)^T h_t^d \tag{13}$$

$$s(h_j^e, h_t^d) = (h_j^e)^T W h_t^d \tag{14}$$

$$s(h_j^e, h_t^d) = v^T tanh(W(h_j^e \oplus h_t^d) + b) \tag{15}$$

As Equações (14) e (15) são as mais utilizadas na tarefa de sumarização abstrativa (SHI et al., 2018).

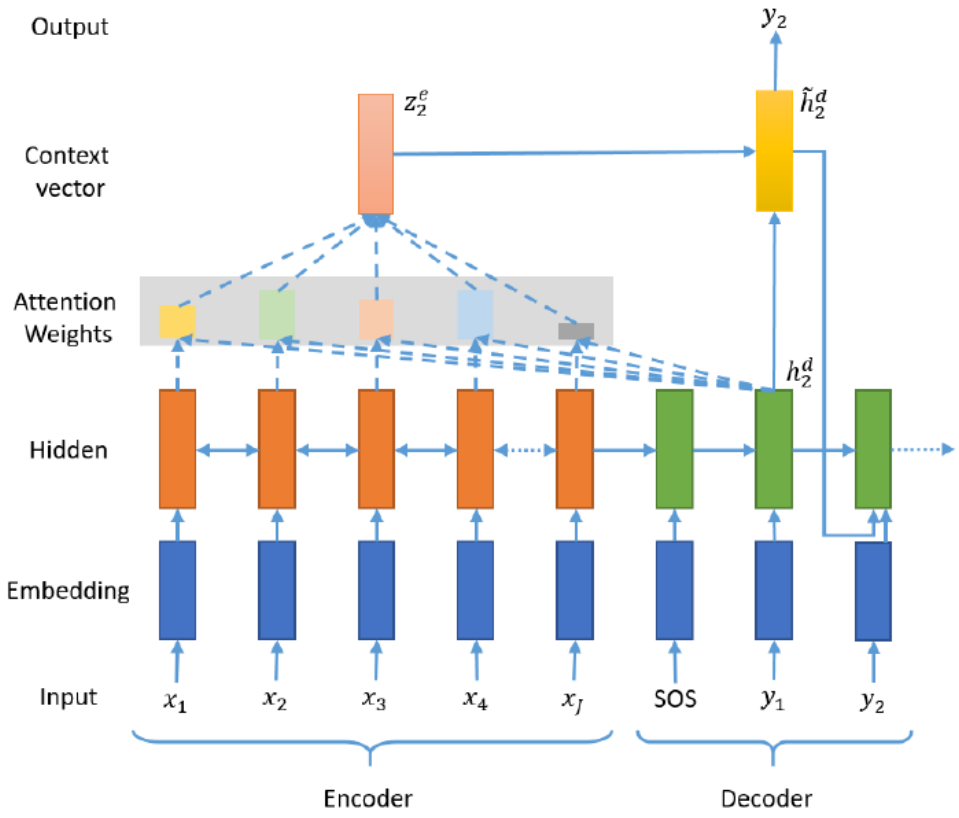
Com a distribuição da atenção, é possível definir o vetor de contexto da palavra alvo como na Expressão 16:

$$z_t^e = \sum_{j=1}^{J} \alpha_{tj}^e h_j^e \tag{16}$$

e o vetor de atenção é obtido a partir da distribuição de atenção obtida por meio da Equação (16) e da camada oculta atual do decodificador h_t^d, por meio da Equação (17):

$$\tilde{h}_t^d = W_z(z_t^e \oplus h_t^d) + b_z \tag{17}$$

Figura 8 – Arquitetura Seq2Seq baseada em atenção



Fonte: SHI et al., 2018

3.3 WORD EMBEDDING

A representação de palavras é estudada há muito tempo, inicialmente, por técnicas não neurais (DEERWESTER et al., 1990; BROWN et al., 1992; SCHUTZE, 1992; LUND; BURGESS, 1996; ANDO; ZHANG, 2005; BLITZER; MCDONALD; PEREIRA, 2006) e por técnicas neurais (BENGIO et al., 2003; COLLOBERT; WESTON, 2008; MIKOLOV et al., 2013c; PENNINGTON; SOCHER; MANNING, 2014; BOJANOWSKI et al., 2017; DEVLIN et al., 2018). O *Word Embedding* é uma representação vetorial contínua, capaz de capturar informações sintáticas e semânticas de palavras (ROSSIELLO; BASILE; SEMERARO, 2017). Os *embeddings* de palavras são colocados em um espaço de alta dimensão, onde os *embeddings* de palavras semelhantes ou relacionadas estão próximos uns dos outros. Da mesma forma, os *embeddings* de palavras não semelhantes são colocados longe uns dos outros (ALVAREZ; BAST, 2017).

O trabalho inovador desenvolvido em (MIKOLOV et al., 2013c) propôs dois simples modelos log-linear que superou todas as arquiteturas anteriores e reduziu drasticamente a complexidade de tempo. Devido à escalabilidade dos *embeddings* tornou-se possível o uso de maiores volumes de dados, o que tornou os *embeddings* muito mais precisos, permitindo o uso de forma confiável para qualquer modelo de Processamento de Linguagem Natural (PLN) (ALVAREZ; BAST, 2017).

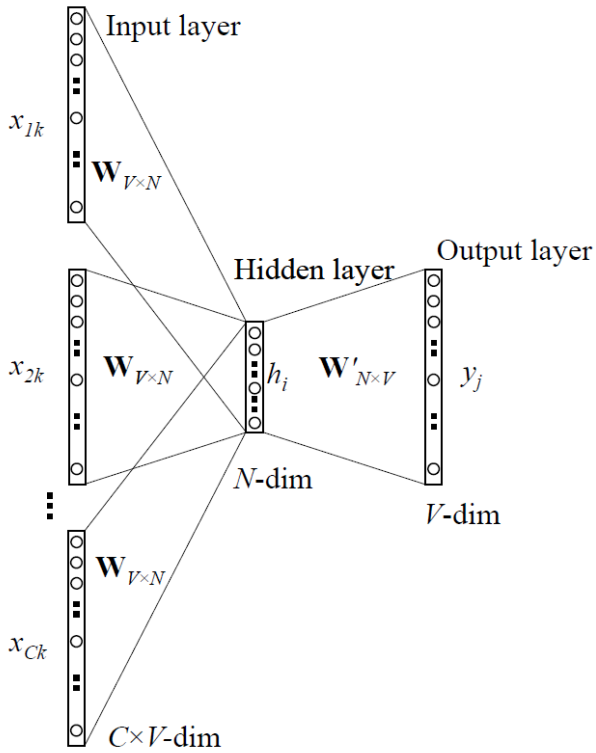
Na Subseção 3.3.1 será apresentado o *Word2Vec* proposto por Mikolov, enquanto que na Subseção 3.3.2 é apresentada uma técnica para redução de dimensão de *Word Embedding*.

3.3.1 Word2Vec

O *Word2Vec* é um modelo proposto em (MIKOLOV et al., 2013c) composto de duas arquiteturas baseadas em redes neurais, CBOW e *Skip-Gram*, utilizado para o aprendizado das representações de palavras. Os modelos anteriores ao *Word2Vec* possuíam uma complexidade computacional alta causada pela não linearidade dos modelos de redes neurais nas camadas ocultas. Por este motivo, o modelo proposto tenta minimizar esse problema de não linearidade.

No modelo CBOW (*Continuous Bag-of-Words*), apresentado na Figura 9, é prevista a partir de um contexto uma palavra central, sendo o contexto interpretado como uma sentença. Para otimizar o modelo é feita a alteração da arquitetura da rede neural, sendo retirada a camada oculta e utilizando a camada de projeção de maneira compartilhada com todas as palavras. Para definir a palavra central prevista, é calculada a média entre todos os vetores de contexto presentes na camada de entrada.

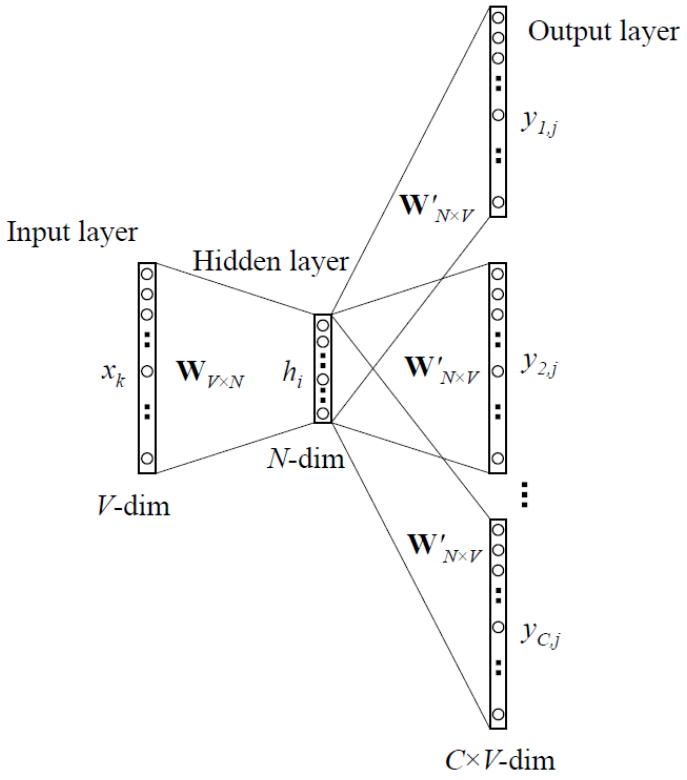
Figura 9 – Modelo Word2Vec: arquitetura CBOW.



Fonte: RONG, 2014

Como pode ser visto na Figura 10, no modelo *Skip-gram* uma palavra é dada com o objetivo de prever o seu contexto, assim, cada palavra atual é utilizada como entrada da rede, sendo essa composta de uma camada de projeção contínua, que por sua vez prevê as palavras do contexto dentro de um determinado intervalo antes e depois da palavra atual. Esse método possui uma complexidade computacional alta, pois a predição do contexto envolve um conjunto de palavras maior, comparado ao CBOW.

Figura 10 – Modelo Word2Vec: arquitetura *Skip-gram*.



Fonte: RONG, 2014

Com esse modelo, é possível obter diferentes tipos de similaridade entre palavras, como por exemplo, a palavra “pequeno” é similar a “menor” assim como “grande” é similar a “maior”. Além disso, é possível obter respostas analisando a relação semântica entre as palavras, como por exemplo a relação entre uma cidade e um país: São Paulo é para o Brasil como Nova Iorque é para os EUA. Um outro exemplo clássico do modelo é a operação dos vetores: $Rei - Homem + Mulher \approx Rainha$, que mostra a qualidade das similaridades obtidas pelo modelo.

3.3.1.1 Arquitetura Skip-Gram detalhada

Em (MIKOLOV et al., 2013a) é descrito de forma detalhada como funciona o modelo *skip-gram*. Esse modelo recebe como entrada uma sequência de palavras definidas como $w_1, w_2, w_3, ..., w_T$, onde os pesos são representados pela matriz W de dimensão T x N, sendo N o número de unidades de projeção e T o tamanho do vocabulário. Para prever o contexto da palavra central w_t, o objetivo do modelo é maximizar as probabilidades médias, como definida na Expressão (18) a seguir:

$$\frac{1}{T}\sum_{i=1}^{T}\sum_{-m\leq j\leq m, j\neq 0}\log p(w_{t+1} \mid w_t) \tag{18}$$

sendo m o raio de palavras do contexto da palavra central a serem analisadas. A versão simplificada de $P(w_{t+1} \mid w_t)$ da Expressão (18) é definida como mostra a Expressão (19):

$$p(c \mid p) = \frac{exp(v_c \cdot v_p)}{\sum_{w=1}^{T} exp(v_w \cdot v_c)} \tag{19}$$

sendo v_c o vetor de contexto, v_p o vetor da palavra central e T é o número de palavras no vocabulário. Uma vez que essa abordagem é computacionalmente custosa, utiliza-se o *softmax* hierárquico, uma abordagem com complexidade computacional menor.

Como citado anteriormente, é definido um raio m na Equação (18) que determina quantas palavras serão analisadas entorno da palavra central. Essa análise é a maximização da Equação (18). Porém, para reduzir o número de amostras analisadas e possíveis ruídos, é então selecionada uma nova palavra central para o contexto da palavra central atual, a fim de minimizar a Equação (18). Por outro lado, com objetivo de tratar a desigualdade entre palavras raras e palavras frequentes no conjunto de treinamento, é utilizada a função sub-amostral definida na Equação (20) a seguir:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{20}$$

onde cada palavra w_i é descartada no conjunto de dados de treino, sendo $f(w_i)$ a frequência da palavra w_i e t um limiar definido. Essa função é utilizada por selecionar sub-amostras de palavras cujas frequências são maiores que t.

3.3.2 Redução de Dimensão

Com objetivo de melhorar a qualidade dos *word embeddings*, em (RAUNAK; GUPTA; METZE, 2019) é proposto um método que combina, de forma eficiente, a Análise de Componentes Principais (PCA) com o Algoritmo de Pós-Processamento (PPA) proposto em (GONG et al., 2018) para construir *word embeddings* de menores dimensões de forma efetiva.

O algoritmo PPA basea-se nas observações geométricas de que o *word embedding* possui um grande vetor médio e boa parte de suas principais informações, depois de substrair o *embedding* por seu vetor médio, estão alocadas em um subespaço de 8 dimensões. Uma vez que todos os *embeddings* compartilham um vetor médio comum e todos os *embeddings* possuem as

mesmas direções dominantes, as quais influenciam fortemente as representações, eliminar esse vetor médio torna os embeddings mais fortes. Esse método é apresentado no Algoritmo 1.

Algoritmo 1 – Algoritmo de Pós-Processamento (PPA)

Entrada: Matriz *word embedding* X e parâmetro limiar D
Saída: Matriz *word embedding* pós-processada X
$X = X - \overline{X}$ // Subtrai a média do *embedding*
$u_i = PCA(X)$ // Calcula as componentes do PCA
para cada $v \in X$ **faça**
  $v = v - \sum_{i=1}^{D}(u_i^T \cdot v)u_i$ // Remove os componentes Top-D
fim

O método proposto por Raunak aplica inicialmente o PPA no *word embedding* original, para tornar o *embedding* mais discriminativo, ou seja, que os padrões dos vetores sejam diferentes entre si. Na sequência, é criada uma representação de baixa dimensão do resultado obtido a partir do PPA utilizando PCA baseado na técnica de redução de dimensão. Por fim, é aplicado novamente o PPA, pois foi observado que, mesmo o PCA tendo sido aplicado em *word embeddings* resultantes do PPA, a variância é explicada de forma desproporcional por alguns componentes após aplicar o PCA. Por esse motivo é aplicado novamente o PPA. O método completo é mostrado no Algoritmo 2.

Algoritmo 2 – Algoritmo proposto por Raunak

Entrada: Matriz *word embedding* X e parâmetro limiar D
Saída: Matriz *word embedding* com dimensão reduzida X
$PPA(X,D)$ // Aplica o Algoritmo 1 (PPA)
$PCA(X)$ // Transforma X utilizando PCA
$PPA(X,D)$ // Aplica o Algoritmo 1 (PPA)

3.4 LIDA

O *Learning Intelligent Decision Agent* (LIDA) é um modelo computacional conceitual e parcialmente implementado que procura cobrir grande parte da cognição humana. A arquitetura LIDA é uma extensão do modelo IDA, um software agente inteligente, automático, "conscience" (BAARS, 1988, 1997; FRANKLIN, 2003) que foi aplicado na Marinha dos Estados Unidos utilizando inteligência artificial, o qual foi desenvolvido pela Universidade de Memphis (FRANKLIN, 2003). O LIDA, inicialmente *learning* IDA, partindo do IDA, adicionou três módulos de aprendizado: perceptivo, episódico e processual. Entretanto, sua evolução foi além

disso e o modelo tornou-se mais genérico. É possível verificar sua evolução em (FRANKLIN; PATTERSON JR, 2006; FRANKLIN et al., 2007, 2014, 2016; KUGELE; FRANKLIN, 2020b).

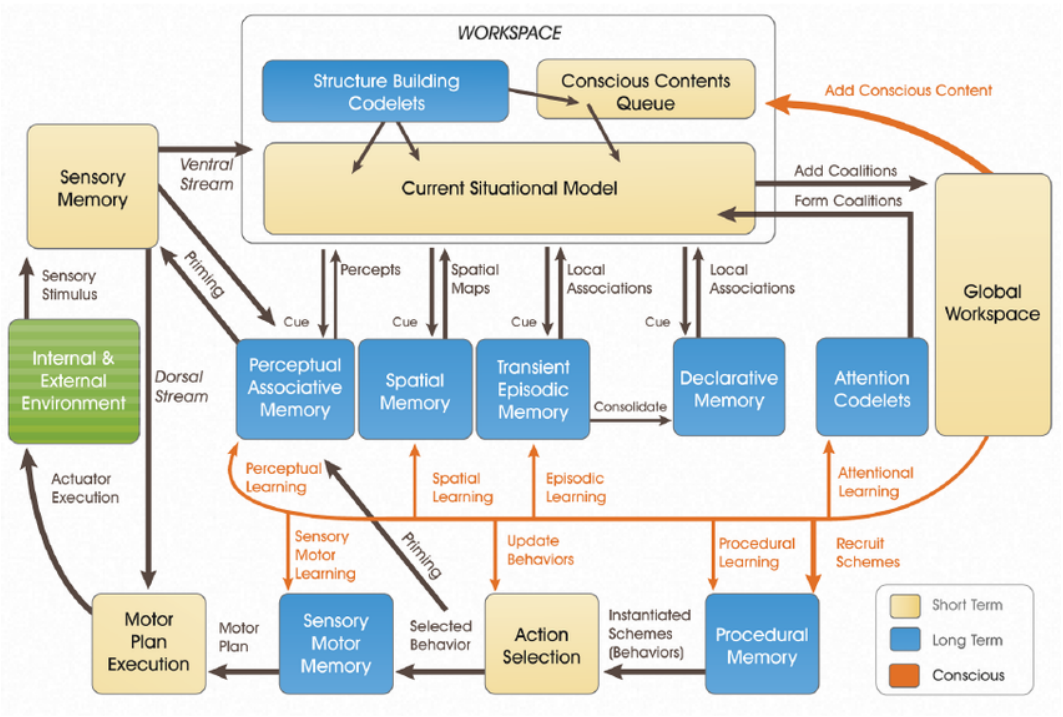
O LIDA basea-se na teoria de (BAARS, 1997) do *Global Workspace* (GWT), uma teoria conceitual sobre o papel da consciência (especificamente do componente de atenção) na cognição. Originalmente criado como um modelo neuropsicológico de processos conscientes e inconscientes, o GWT é, portanto, uma teoria de como a consciência funciona dentro da cognição. Além do GWT, o modelo LIDA é fundamentado em uma série de teorias psicológicas e neuropsicológicas, incluindo: sistemas de símbolos perceptivos, memória de trabalho, memória de provisões, memória de trabalho de longo prazo, entre outras.

A seguir serão apresentados o ciclo cognitivo e os módulos do modelo conceitual do LIDA baseado em (FRANKLIN et al., 2016).

3.4.1 Ciclo cognitivo do LIDA

O ciclo cognitivo do LIDA é descrito em módulos (Figura 11), os quais interagem uns com os outros para acessar os elementos de dados necessários. Essa interação entre os módulos não ocorre serialmente, ou seja, o modelo LIDA não executa suas funções de forma serial, ocorrendo apenas para o processo de consciência e para a seleção de ação, os demais processos e módulos operam de forma paralela.

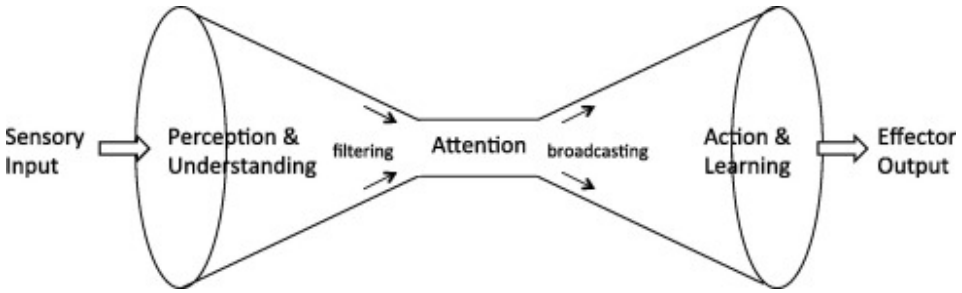
Figura 11 – Ciclo cognitivo em módulos do modelo LIDA. Em cor bege claro, são destacados os módulos que possuem memória de curto prazo e em cor azul, são destacados os módulos que possuem memória de longo prazo. As conexões em cor preta demonstram o fluxo de informação entre os módulo e as conexões em cor vermelha demonstram o fluxo de aprendizado entre os módulos do modelo. Por fim, na cor verde é destacado o ambiente externo e interno, onde são criados os estímulos de entrada e as ações de execução do modelo como saída.



Fonte: FRANKLIN et al., 2016.

O ciclo cognitivo do LIDA (Figura 11) pode ser dividido em três grandes fases (Figura 12): a fase de percepção e entendimento, a fase de atenção e a fase de ação e aprendizado.

Figura 12 – Diagrama esquemático das 3 fases do ciclo cognitivo do modelo LIDA.



Fonte: FRANKLIN et al., 2016.

Utilizando os dados de entrada, a fase de percepção e entendimento (Figura 12) atualiza a compreensão da situação atual. Isso ocorre por meio do estímulo sensorial, tanto externo

quanto interno, enviado para a Memória Sensorial (Figura 11) onde é representado. O conteúdo gerado pela Memória Sensorial (Figura 11) envolve a Memória Associativa Perceptiva (Figura 11) e o Modelo Situacional Atual (Figura 11). A Memória Associativa Perceptiva (Figura 11) serve como uma memória de reconhecimento que produz uma percepção disponibilizada para o Modelo Situacional Atual (Figura 11). Utilizando o conteúdo perceptivo e de entrada, adicionalmente ao conteúdo restante que ainda não decaiu, o Modelo Situacional Atual (Figura 11) se atualiza continuamente por meio da Memória Associativa Perceptiva (Figura 11), Memória Espacial (Figura 11), Memória Transiente Episódica (Figura 11) e Memória Declarativa (Figura 11), fazendo uso do retorno das associações locais. Atualizações adicionais são realizadas pelos Codeletes Construtores de Atenção (Figura 11), utilizando conteúdo do Modelo Situacional Atual (Figura 11) e da Fila de Conteúdo Consciente (Figura 11).

A fase de atenção (Figura 12) filtra o conteúdo de entendimento da fase de percepção e entendimento por importância, distribuindo assim o conteúdo consciente de forma global de acordo com a teoria do GWT. Para isso, cada Codelete de Atenção (Figura 11) de sentinela, analisa continuamente o Modelo Situacional Atual (Figura 11) em busca de conteúdo que possa se tornar consciente. Ao encontrar algo, cria uma aliança, que no Espaço de Trabalho Global (Figura 11) junta-se a uma competição pela consciência. A vencedora e a mais relevante aliança tem seu conteúdo distribuído globalmente, pela qual torna-se consciente.

Finalmente, a fase de ação e aprendizado (Figura 12) seleciona e executa uma resposta apropriada, além de aprender por meio dos sistemas de memória, permitindo que esses módulos selecionem a parte do conteúdo consciente do ciclo mais adequada para aprender. Para isso, a Memória Processual (Figura 11), que define o que se deve fazer e quando, utiliza o conteúdo consciente para instanciar comportamentos adequados aos estímulos recebidos. O módulo de Seleção de Ação (Figura 11) escolhe um desses comportamentos que são enviados à Memória Sensorial Motora (Figura 11) para a criação ou seleção do planejamento motor, para que seja executado.

3.4.2 Memória Sensorial

A Memória Sensorial (Módulo *Sensory Memory* na Figura 11) recebe estímulos vindos do ambiente externo por meio de sensores e os transforma em representações. O resultado é passado simultâneamente para os módulos da Memória Associativa Perceptiva (Figura 11) e do

Modelo Situacional Atual (Figura 11). Essa memória é de curto prazo, portanto as informações são armazenadas por um curto período de tempo.

3.4.3 Memória Perceptiva Associativa

A Memória Perceptiva Associativa (módulo *Perceptual Associative Memory* na Figura 11), também conhecida como PAM, é uma memória de reconhecimento a qual torna as representações vindas da Memória Sensorial (Figura 11) em percepções. Para isso, a PAM utiliza nós e *links*, com os nós representando objetos, ações, eventos, além dos *links* representando as relações entre os nós. Cada nó tem um nível de ativação base, além disso, que podem ser intensificados por meio de sugestões do Modelo Situacional Atual (Figura 11) que ainda recebe cópias de estruturas que possuem níveis de ativação muito altos. Cada distribuição de conteúdo consciente feita pelo Espaço de Trabalho Global (Figura 11) oferece à PAM a oportunidade de aprender novas entidades e reforçar as relações das mesmas. A PAM é uma memória de longo prazo que mantém as informações por um longo período de tempo e suas entidades decaem regularmente.

3.4.4 Memória Espacial

A Memória Espacial (módulo *Spatial Memory* na Figura 11) é a parte do sistema de memórias que codifica, guarda e lembra de informações espaciais do ambiente e de orientações do agente. No LIDA, representações espaciais são primeiro construídas no Espaço de Trabalho. Além das relações das entidades reconhecidas, representadas pelas estruturas da PAM, suas posições relativas ao agente podem ser obtidas de forma perceptiva. Essas posições relativas são representadas como "Vetor Espacial Egocêntrico" entre a própria representação e a representação do objeto, que são como uma espécie de *links* da PAM que contém informações de posição. Além disso, existe um *grid* espacial alocêntrico – um *grid* de nós de lugares da PAM representando locais específicos do ambiente – construídos e mantidos pelos Codeletes Construtores de Estruturas Espaciais (Figura 11). Além de atualizar e manter os *links* egocêntricos, esses codeletes também conectam objetos percebidos aos seus nós de lugares correspondentes, e atualiza essas conexões durante o movimento.

As representações egocêntricas específicas são transientes e temporárias, e não precisam ser guardadas a longo prazo. Por sua vez, as representações alocêntricas, quando se tornam

conscientes, são armazenadas na Memória Espacial (Figura 11), sendo essa uma das memórias de longo prazo do LIDA. Por outro lado, memórias espaciais de longo prazo também são utilizadas sempre que objetos relevantes aparecem no Espaço de Trabalho (Figura 11), ajudando a recuperar mapas alocêntricos encontrados anteriormente.

3.4.5 Memória Episódica Transiente

A memória episódica é uma memória para eventos que representam episódios situacionais (o que?), temporais (quando?) e de localização (onde?). A versão do modelo LIDA é de curto prazo, Memória Episódica Transiente (módulo *Transient Episodic Memory* na Figura 11), nela, as memórias não reforçadas duram por um determinado período. Além disso, novos eventos podem ser aprendidos por meio das distribuições de conteúdo consciente e os eventos mais antigos podem ser reforçados. Também, os eventos podem decair e aqueles que, por um determinado período não decaíram, são consolidados na Memória Declarativa (Figura 11), funcionando de maneira semelhante ao que ocorre na memória dos humanos durante o sono (PRASAD; STARZYK, 2010; TONONI, 2012).

3.4.6 Memória Declarativa

A Memória Declarativa (módulo *Declarative Memory* na Figura 11) é a versão de longo prazo da Memória Episódica Transiente (Figura 11). Nela, ao invés do aprendizado ocorrer por meio da distribuição de conteúdo consciente, como em outras memórias no LIDA, ocorre via consolidação da Memória Episódica Transiente (Figura 11). Nesse período de consolidação, qualquer memória que não tiver decaído na Memória Episódica Transiente (Figura 11) será consolidado na Memória Declarativa (Figura 11). Essa consolidação inclui a criação de traços de memória para novos eventos e reforço de traços de eventos passados recém chegados à Memória Episódica Transiente e que não decaíram.

A Memória Declarativa possui dois tipos de memória: a Memória Autobiográfica e a Memória Semântica. A Memória Autobiográfica é o tipo de memória de eventos completos, possuindo as informações situacionais, temporais e de localização sem perdas. Por sua vez, a Memória Semântica armazena memórias com perda de traços temporais e de localização. Assim, o que restar da memória é mantido na forma de fatos, regras etc.

3.4.7 Espaço de Trabalho

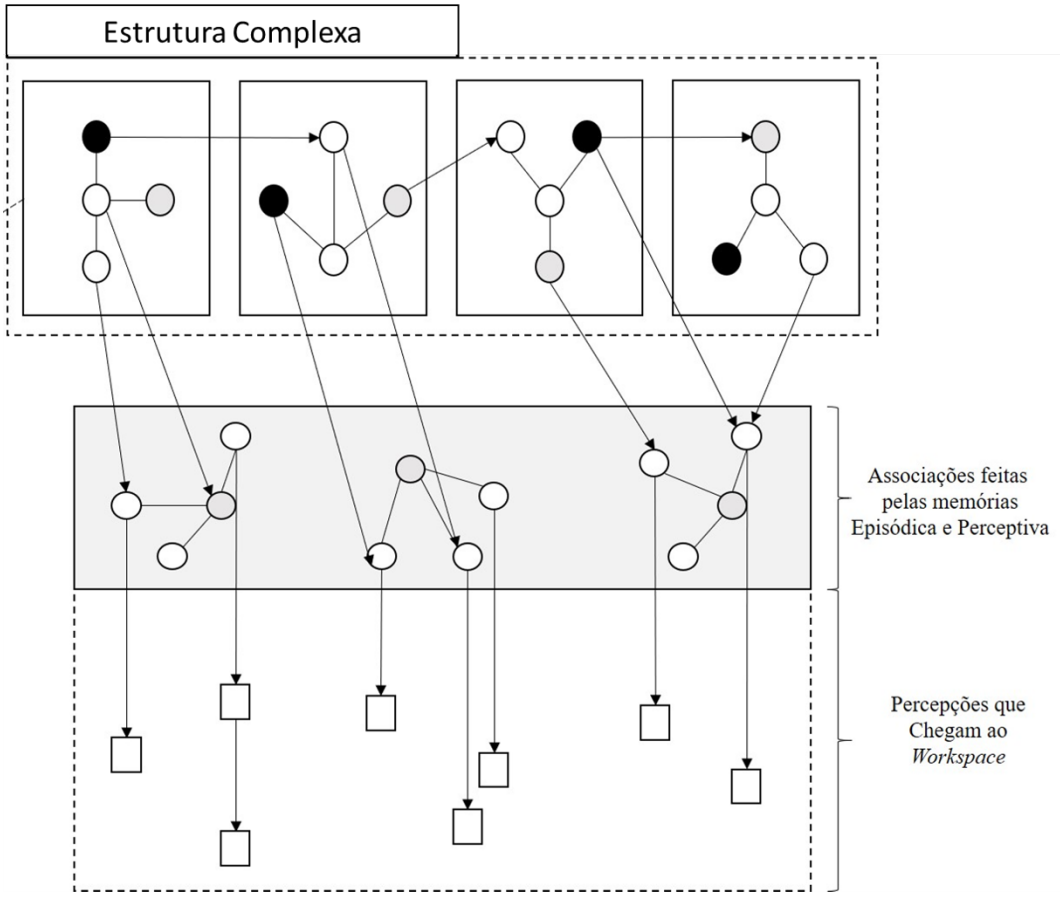
O Espaço de Trabalho (módulo *Workspace* na Figura 11) é uma memória de curto prazo formada pelos módulos: Modelo Situacional Atual (Figura 11), Codeletes Construtores de Estruturas (Figura 11) e Fila de Conteúdo Consciente (Figura 11), que serão descritos a seguir.

3.4.7.1 Modelo Situacional Atual

O Modelo Situacional Atual (módulo *Current Situacional Model* na Figura 11), também conhecido como CSM, recebe repetidamente informações sensoriais externas e internas, ambas diretamente da Memória Sensorial (Figura 11) e das percepções da PAM. O CSM se atualiza de forma contínua para acompanhar a situação atual do LIDA. A informação da PAM chega na forma de estruturas de *links* e nós, enquanto que a informação vinda da Memória Sensorial (Figura 11) deve ser convertida na forma de *links* e nós pelos Codeletes Construtores de Estruturas (Figura 11).

As estruturas que chegam ao CSM automaticamente consultam cada uma das memórias de longo prazo, PAM, Memória Espacial (Figura 11), Memória Episódica Transiente (Figura 11) e Memória Declarativa (Figura 11), o que pode resultar em associações locais. Cada uma dessas associações locais são novas estruturas de entrada, o que pode gerar uma nova consulta ao sistema de memórias, a qual pode produzir novas associações. Sendo assim, novas estruturas são adicionadas continuamente ao CSM. Simultaneamente, as estruturas decaem em taxas variáveis em dezenas de segundos. Em um determinado momento, o conteúdo do CSM pode ser extenso e complexo como pode ser visto na Figura 13.

Figura 13 – Estrutura complexa formada no CSM devido às constantes consultas e associações às demais memórias.



Fonte: adaptado de FRANKLIN et al., 2016.

3.4.7.2 Codeletes Construtores de Estruturas

Os Codeletes Construtores de Estruturas (módulo *Structure Building Codelets* na Figura) criam estruturas para o CSM e possuem a habilidade de reconhecer relações entre conceitos e objetos, como por exemplo, similaridade, casualidade etc. Os Codeletes Construtores de Estruturas (Figura 11) monitoram continuamente o CSM e a Fila de Conteúdo Consciente (Figura 11) a procura de conteúdo de seu interresse. Se esse conteúdo é encontrado, os codeletes executam uma ação que resulta em modificações no CSM. As possíveis ações incluem criar novas associações (*links*), criar novo conteúdo, como nós de categorias, ou remover associações anteriores e conteúdo.

Os Codeletes Construtores de Estruturas (Figura 11) possuem nível de ativação próprio. Além disso, devem atribuir uma ativação atual para cada nova estrutura que eles criam. Essa ativação é uma função das ativações atuais das diversas estruturas pré-existentes de interesse

no CSM. Além disso, quando um Codelete Construtor de Estruturas (Figura 11) reconhece o conteúdo que ele construiu via distribuição de conteúdo consciente, esse codelete recebe um aumento pequeno na sua ativação. Como resultado, os Codeletes Construtores de Estruturas (Figura 11) que criam consistentemente estruturas que são "úteis" e terão ativações maiores. Paralelamente, os codeletes que criarem estruturas menos úteis irão perder lentamente nível de ativação, e eventualmente podem ser descartados.

3.4.7.3 Fila de Conteúdo Consciente

A Fila de Conteúdo Conciente (módulo *Conscious Contents Queue* na Figura 11) é uma memória de curto prazo que guarda as últimas dezenas de conteúdos consciente. Uma nova distribuição de conteúdo consciente é adicionada ao fim da Fila, empurrando para fora o conteúdo consciente da frente. A Fila de Conteúdo Consciente (Figura 11) deixa de ser uma fila apenas para os Codeletes Construtores de Estruturas (Figura 11), os quais podem utilizar os dados em qualquer ponto da Fila, e não apenas do que está na frente. A Fila também pode ser utilizada para estimar a duração de eventos, contando quantas distribuições conscientes que ocorreram anteriormente - guardadas na Fila - contém um evento, e as percepções centrais.

3.4.8 Codeletes de Atenção

Os Codeletes de Atenção (módulo *Attention Codelets* na Figura 11) são responsáveis por implementar o filtro de atenção por relevância. Como visto na Subseção 3.4.7.1, em um determinado momento, o conteúdo do CSM pode ser extenso e complexo, o que pode ser difícil de lidar. Sendo assim, os Codeletes de Atenção (Figura 11), assim como os Codeletes Construtores de Estruturas (Figura 11), monitoram continuamente o CSM em busca de alguma estrutura relevante que correspondam ao interresse particular dos Codeletes (Figura 11). Ao encontrar alguma estrutura relevante no CSM, o codelete o incorpora em uma aliança (pode ser composta de mais de uma estrutura e pode criar uma aliança conjunta com outros Codeletes de Atenção (Figura 11)), que é então movida ao Espaço de Trabalho Global (Figura 11) para competir pela consciência.

Os Codeletes de Atenção (Figura 11) devem atribuir uma ativação para a nova aliança, com base no qual competirá pela consciência. O valor dessa ativação depende de quatro fatores: da ativação das estruturas incorporadas na aliança: do nível de ativação do codelete de atenção:

do quanto as estruturas correspondem ao interesse particular dos Codeletes e do período refratário ao qual uma aliança vencedora com um determinado alto nível de ativação põe os Codeletes de Atenção (Figura 11), sendo esse período uma espécie de espera, podendo afetar estruturas relevantes de serem levadas para competição.

Existem quatro tipos de Codeletes de Atenção (Figura 11), sendo eles: Codelete de Atenção Padrão, que seleciona a estrutura mais relevante no CSM pelo seu grau de ativação; Codelete de Atenção Específica, que busca conteúdos específicos que foram aprendidos por eles; Codeletes de Expectativa, onde são buscados resultados de suas próprias ações; e os Codeletes de Intenção, onde são buscados conteúdos que podem ajudar o modelo a atingir seu objetivo.

3.4.9 Espaço de Trabalho Global

No Espacço de Trabalho Global (módulo *Global Workspace* na Figura 11) as alianças movidas pelos Codeletes de Atenção competem para ter suas estruturas distribuídas para grande parte do modelo LIDA. A competição ocorre de maneira simples, a aliança com maior ativação vence. Porém, a competição não é assíncrona e não opera com *clock*. É necessário ter algo que inicie a competição. Para isso, existem quatro gatilhos diferentes. O primeiro é um simples limite de ativação, ao chegar uma nova aliança que possua um nível de ativação acima de um determinado limite uma competição é iniciada. Outro gatilho é acionado no momento em que a soma de ativações das alianças presentes no Espaço de Trabalho Global (Figura 11) excede um determinado limite. Uma competição pode ser iniciada também caso nenhuma aliança chegue ao Espaço de Trabalho Global (Figura 11) por um determinado limite de tempo. Por fim, existe o gatilho padrão, onde uma competição é iniciada após passar um período de tempo sem que tenha ocorrido uma propagação consciente.

3.4.10 Memória Processual

A Memória Processual (módulo *Procedural Memory* na Figura 11) é a memória que define o que fazer em determinada cirscunstância para atingir um objetivo. A estrutura de dados da Memória Processual (Figura 11) é o *scheme*, que consiste em um contexto, uma ação, um resultado e um nível de ativação. Ao receber uma distribuição de conteúdo consciente, qualquer *scheme*, cujo contexto aumenta significativamente com o conteúdo distribuído, instancia uma

cópia de si mesmo, chamada de comportamento, com suas variáveis especificadas de acordo com o conteúdo consciente. A ativação atribuída ao comportamento instanciado depende da ativação do conteúdo consciente, do nível de ativação do *scheme*, do grau de similaridade entre o conteúdo consciente e o contexto do *scheme* e da proximidade do resultado do *scheme* com o objetivo do modelo. O comportamento instanciado é então passado para o módulo de Seleção de Ação (Figura 11).

3.4.11 Seleção de Ação

Na Seleção de Ação (módulo *Action Selection* na Figura 11) o comportamento instanciado recebido da Memória Processual (Figura 11) passa a ser nó de uma rede de comportamento. Por ser um dos poucos módulos assíncronos, a Seleção de Ação (Figura 11) necessita de um esquema de deliberação especial para decidir quando deve-se iniciar o processo. Existem 3 tipos de condições para iniciar esse processo: a ativação de um comportamento estar acima de um determinado nível, a ativação total de todos os comportamentos na rede de Seleção de Ação (Figura 11) estar acima de determinado nível, e quando nenhuma ação é executada em determinada quantidade de tempo. Quando o processo de deliberação é iniciado, o comportamento que melhor se encaixar no contexto atual é selecionado e, então, enviado para a PAM, caso seja um comportamento interno, ou para a Memória Sensorial Motora (Figura 11), caso seja externo. O envio do comportamento à PAM é chamado de *Priming*.

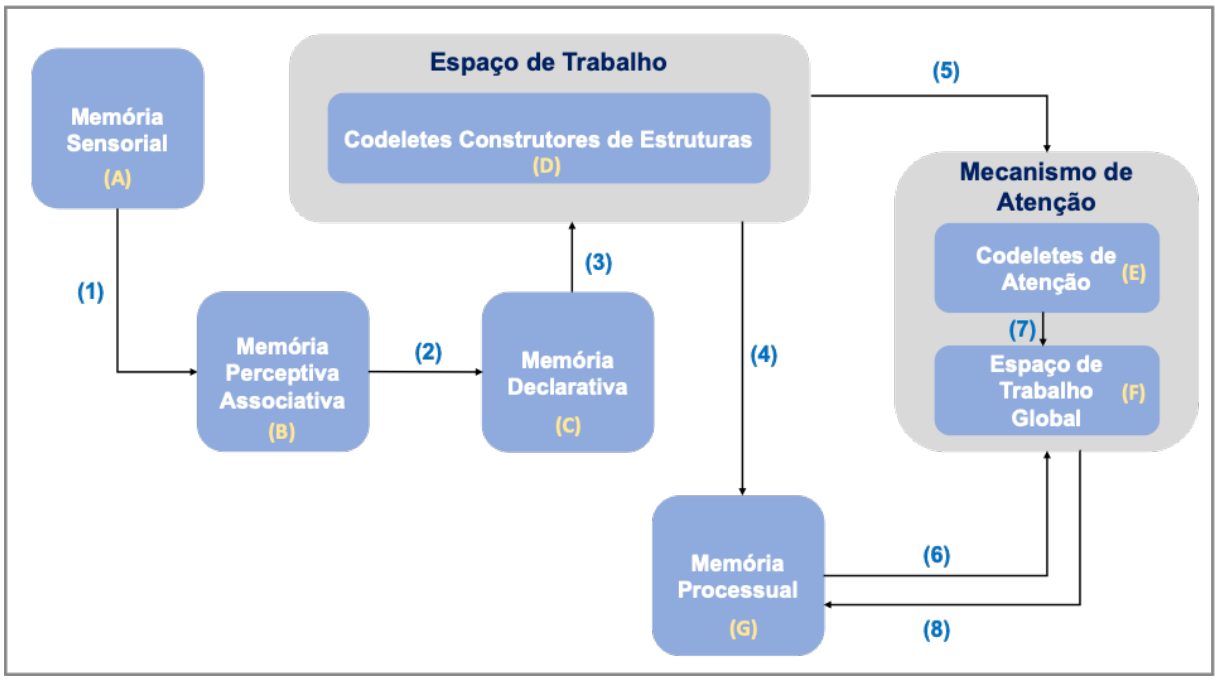
3.4.12 Memória Sensorial Motora e Executor de Plano Motor

A Memória Sensorial Motora (módulo *Sensory Motor Memory* da Figura 11), por sua vez, seleciona um plano motor que realize esse comportamento e o envia para o Executor de Plano Motor (Figura 11), para que ele seja devidamente posto em prática. O plano motor é uma estrutura de dados baseada na arquitetura de subsunção, um tipo de mecanismo de controle de motor reativo, onde os dados sensoriais são conectados diretamente com comandos motores. Os comandos gerados são enviados ao Executor de Plano Motor (módulo *Motor Plan Execution* da Figura 11), para que o agente possa agir no ambiente de acordo com o que foi percebido no início do ciclo cognitivo.

4 METODOLOGIA PROPOSTA

A metodologia proposta nesse trabalho é um modelo de sumarização de texto baseado no modelo teórico cognitivo do LIDA, descrito na Seção 3.4. Como pode ser observado na Figura 43, o modelo possui 7 módulos e 8 conexões. Cada um dos módulos será detalhado nas seções a seguir, assim como o fluxo de dados por meio das conexões do modelo.

Figura 14 – Modelo proposto de sumarização de texto baseado no modelo teórico cognitivo LIDA. Em amarelo, são apresentadas as letras que se referem aos módulos do modelo, e em azul, são apresentados os números que se referem às conexões entre os módulos.



Fonte: autora.

4.1 BASES DE DADOS

Para desenvolver o modelo proposto, será utilizada a base de dados *CNN/Daily Mail* proposta em (HERMANN et al., 2015). Neste trabalho, Hermann apresenta o conjunto de dados *CNN/Daily Mail* com o objetivo de aplicar na tarefa de compreensão de leitura. Para isso, são extraídos dos *websites* da CNN 96 mil e do *Daily Mail* 220 mil artigos. Ambos os *websites* apresentam, em forma de *abstract*, pequenos pontos de resumo em seus artigos; ou seja, não copiam simplesmente as frases dos artigos. (HERMANN et al., 2015) utilizam esses resumos para criarem perguntas com uma entidade substituída, com objetivo de prever essa entidade.

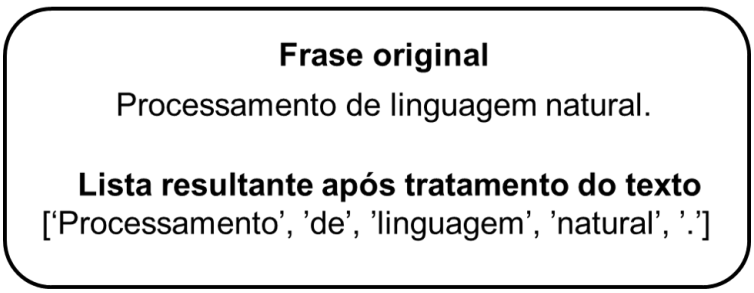
Em (NALLAPATI et al., 2016) é criado um tratamento para esse conjunto de dados, tornando-o possível para aplicação na tarefa de sumarização de texto. Nesse trabalho, foi modificado o código proposto por (HERMANN et al., 2015) para restaurar os resumos na ordem original de modo a se obter resumos de múltiplas sentenças. O conjunto de dados resultante desse tratamento possui 286.817 pares de treinamento, 13.368 pares de validação e 11.487 pares de teste. Os artigos do conjunto de treinamento possuem 766 palavras abrangendo 29,74 sentenças em média, enquanto os resumos possuem 53 palavras e 3,72 sentenças em média.

A presente dissertação utilizará o tratamento executado em (SEE; LIU; MANNING, 2017), que aplica os códigos executados em (NALLAPATI et al., 2016), e obtém os mesmos artigos, porém não mascara os nomes de entidades, retirando essa etapa de pré-processamento.

4.2 MEMÓRIA SENSORIAL

O módulo Memória Sensorial (Figura 43, Bloco A) é responsável por receber os estímulos do ambiente externo e transformá-los em representações, sendo assim, desempenhará a tarefa de pré-processamento do texto. O texto de entrada será *tokenizado* (Figura 15), e ainda, todas as palavras, após serem transformadas em *tokens*, serão convertidas em palavras minúsculas. Por fim, os *tokens* serão enviados para o próximo módulo (Memória Perceptiva Associativa).

Figura 15 – Ilustração do resultado do tratamento de texto após *tokenização*, que resulta na separação das palavras por espaços em branco, e transformação dos *tokens* em minúsculos.



Fonte: autora.

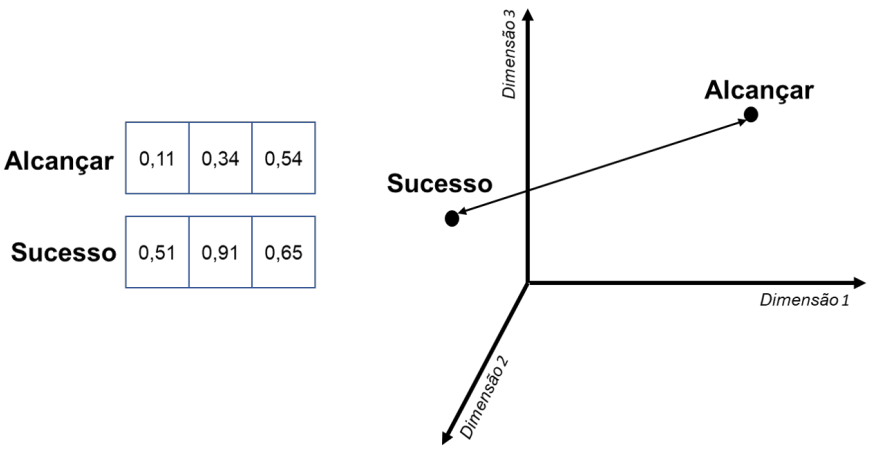
4.3 MEMÓRIA PERCEPTIVA ASSOCIATIVA

O módulo da Memória Perceptiva Associativa (Figura 43, Bloco B) é uma memória de reconhecimento a qual torna as representações (*tokens*) vindas da Memórial Sensorial em percepções que serão criadas utilizando o modelo *word2vec* (Seção 3.3.1), que gera vetores espaciais das palavras, como pode ser observado na Figura 16, permitindo obter a similaridade e as relações semânticas entre elas. Essa abordagem remete aos nós e *links* propostos no modelo teórico do LIDA que representam as relações dos objetos. Os vetores gerados pelo *word2vec* são encaminhados para o módulo seguinte: Memória Declarativa.

4.4 MEMÓRIA DECLARATIVA

O módulo Memória Declarativa (Figura 43, Bloco C) é responsável por fortalecer as relações estabelecidas na Memória Perceptiva Associativa. Para isso, aplica a técnica de redução de dimensão definida na Seção 3.3.2. A Memória Perceptiva Associativa envia a matriz de *Word Embedding* para a Memória Declarativa, que aplica o PPA primeiro, o PCA na sequência e novamente o PPA. Com isso, é obtida a matriz de *Word Embeddings* com sua dimensão reduzida e suas relações fortalecidas, uma vez que o algoritmo intensifica a singularidade de cada palavra, tornando a matriz de *embeddings* mais discriminativa. Por fim, envia a matriz reduzida para os Codeletes Construtores de Estruturas.

Figura 16 – Exemplo dos vetores gerados pelo *Word2Vec* das palavras "Sucesso"e "Alcançar".



Adaptado de: KIM, 2019

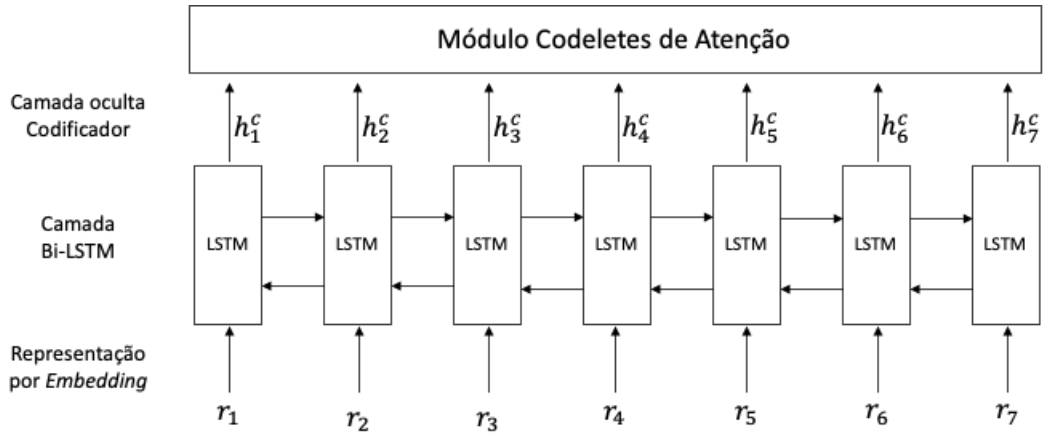
4.5 ESPAÇO DE TRABALHO

O Espaço de Trabalho é composto do módulo Codeletes Construtores de Estruturas. Nele são recebidas as informações vindas da Memória Declarativa, a qual gera a estrutura compactada dessas informações e a envia para os Codeletes de Atenção.

4.5.1 Codeletes Construtores de Estruturas

O módulo Codeletes Construtores de Estruturas (Figura 43, Bloco D) é composto de uma rede RNN-LSTM bidirecional de uma camada (Figura 17) que atuará como codificador dos dados vindos da Memória Declarativa (matriz de *Word Embedding* reduzida). A rede irá gerar o vetor de contexto; ou seja, reconhecerá as relações entre palavras e frases que constituem a totalidade do texto. Para isso, será utilizada a última camada da rede codificadora como contexto do texto, o qual será enviado para o módulo Codeletes de Atenção.

Figura 17 – Codificador RNN-LSTM bidirecional de uma camada



Fonte: autora

4.6 CODELETES DE ATENÇÃO

O módulo Codeletes de Atenção (Figura 43, Bloco E) é responsável por determinar a pontuação das sentenças do texto original a serem utilizadas para a sumarização do texto a cada passo da Memória Processual, responsável pela sumarização. Cada um dos estados da última camada oculta da rede executada pelos Codeletes Construtores de Estruturas é responsável por uma parte do texto original. Os Codeletes de Atenção utilizam os pesos dos estados para cal-

cular a pontuação de atenção de acordo com o passo em que a Memória processual está. A Memória processual utiliza o texto sumarizado de forma supervisionada para determinar os pesos da rede decodificadora. Sendo assim, para cada sentença (passo) da Memória processual, os Codeletes de Atenção calculam a pontuação dos estados dos Codeletes Construtores de Estruturas para determinar qual estado possui maior pontuação para a sentença que a Memória processual está executando. A pontuação é determinada pelo produto entre os estados da camada oculta $h^c = (h_1^c, h_2^c, ..., h_j^c)$ dos Codeletes Construtores de Estruturas e o estado atual da Memória processual s_t, como pode ser visto na Equação (21). Após calcular a pontuação de atenção, o módulo Codeletes Construtores de Atenção a envia para o módulo Espaço de Trabalho Global.

$$p_t = [s_t^T h_1^c, s_t^T h_2^c, ..., s_t^T h_j^c] \tag{21}$$

4.7 ESPAÇO DE TRABALHO GLOBAL

O módulo Espaço de Trabalho Global (Figura 43, Bloco F) recebe a pontuação de atenção p_t dos Codeletes de Atenção e, a partir dessa pontuação, cria a distribuição de probabilidade de atenção α_t. Para isso, aplica a *softmax* em p_t, como pode ser visto na Equação (22). Na sequência, a distribuição α_t é utilizada para determinar os pesos de atenção dos estados gerados pelos Codeletes Construtores de Estruturas. A Equação (23) demonstra essa operação. Por fim, o Espaço de Trabalho Global envia o resultado a_t para a Memória processual.

$$\alpha_t = \frac{exp(p_t)}{\sum_{i=1}^{N} exp(p_{ti})} \tag{22}$$

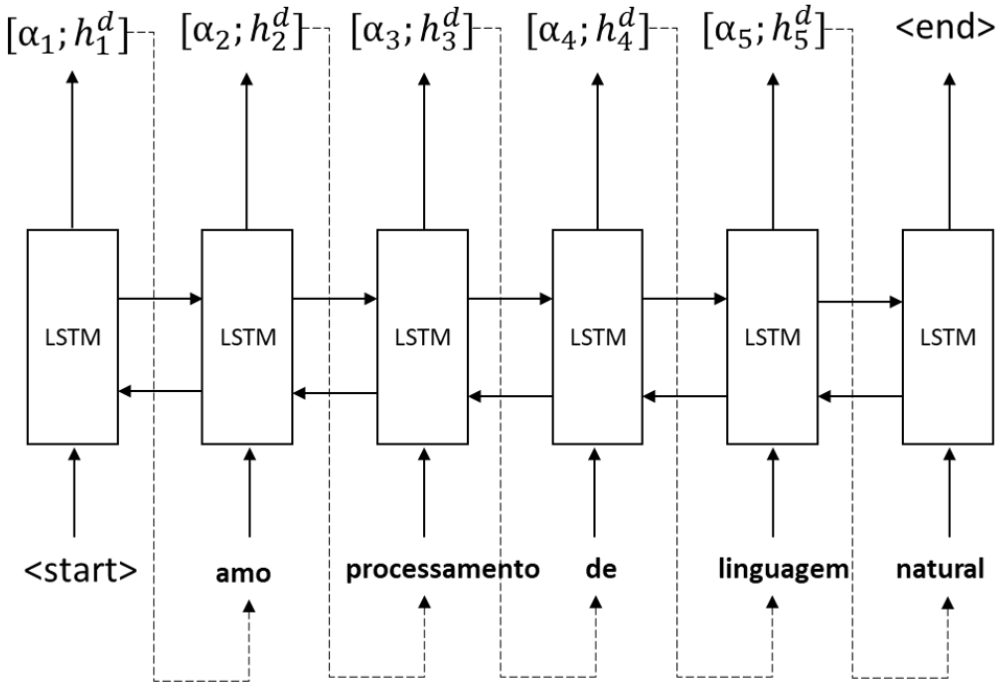
$$a_t = \sum_{i=1}^{N} \alpha_{ti} h_i^c \tag{23}$$

4.8 MEMÓRIA PROCESSUAL

O módulo Memória Processual (Figura 43, Bloco G) é composto de uma rede RNN-LSTM bidirecional de uma camada (Figura 18) que funcionará como um decodificador das informações codificadas pelos Codeletes Construtores de Estruturas. A Memória processual receberá como entrada o vetor com os pesos vindo do Espaço de Trabalho Global. Sendo assim,

irá concatenar esse vetor com o estado atual da camada oculta do decodificador e utilizará esse vetor resultante como entrada da rede e, para cada estado, irá prevê uma palavra que irá compor o resumo.

Figura 18 – Decodificador RNN-LSTM bidirecional de uma camada, sendo $[\alpha_t; h_t^d]$ o vetor resultante da concatenação dos pesos vindos do Espaço de Trabalho Global com o estado atual da camada oculta do decodificador.



Fonte: autora

Figura 19 – Exemplo de um texto e resumo esperado extraídos da base de dados *CNN/Daily Mail.*

Texto Original	Resumo Esperado
'(CNN) -- Wise men say to look before you leap. In Alaska, it\'s advisable to look before you land. That\'s because, in Alaska, where seaplanes are common, you just might land on a whale. Last week in tiny, remote Angoon, Thomas Hamm was shooting video of a seaplane coming in for a landing. It was a mundane scene in the island community that\'s only accessible by boat or seaplane. The video starts out normal. But as the plane lowers, it\'s clear something is different about this approach. "All the sudden, the pilot advanced the throttle and I didn\'t know why. I thought, \'Oh something must be wrong,\'" Hamm told CNN. That something was a whale, a humpback, swimming just under the surface. For a moment, it appeared the whale and plane would collide. But the pilot pulled up, getting just enough lift to avoid the mammal. The plane landed safely seconds later. Later Hamm showed the pilot the video he shot. Hamm said the pilot told him he didn\'t notice the whale; he reacted to the commotion on the shore. Guys were pointing and yelling. Right as the pilot pulled up, the whale breached, clearing his blowhole and drenching the plane\'s windshield. That\'s one way to make a splash. Jetliner diverts to Pacific atoll, mechanical glitch blamed . Rare albino whale \'parades\' off Australian coast .'	"Brandon Stanton launched Humans of New York blog .\nPartnering with U.N., he's on a world tour that started in Iraq and Jordan .\nHis mini narratives reveal the struggles and dreams of ordinary people ."

Fonte: autora

5 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados e discutidos os resultados dos experimentos realizados para análise do comportamento da metodologia proposta utilizando o modelo cognitivo computacional LIDA (Seção 3.4) como arquitetura base para sumarização abstrativa de textos.

5.1 DEFINIÇÃO DO NÚMERO DE DIMENSÕES DO WORD EMBEDDING

No módulo Memória Perceptiva Associativa (Seção 4.3) é utilizado o *Word2Vec* (Seção 3.3.1) para representar as palavras do texto a ser resumido. O *Word2Vec* é um tipo de *Word Embedding*, sendo esse uma representação vetorial contínua das palavras, capaz de capturar informações sintáticas e semânticas. Sendo assim, para analisar o melhor desempenho do *Word Embedding*, foi utilizada a avaliação intrínseca.

A avaliação intrínseca testa a qualidade de uma representação das relações sintáticas e semânticas entre as palavras. Para isso, a métrica de similaridade de palavras correlaciona a distância entre os vetores de palavras gerado pelo *Word Embedding* e a similaridade semântica compreendida por humanos das bases de dados *WordSim353* (AGIRRE et al., 2009), *SimLex-999* (HILL; REICHART; KORHONEN, 2014) e *Google Analogy Test Set* (MIKOLOV et al., 2013b). Essas relações são estabelecidas pela seguinte analogia:

$$a : b :: c :? \tag{24}$$

que significa: a gera b, então c deve gerar d, sendo d o valor gerado pelo *Word Embedding* a ser analisado, representado na Equação (24) como interrogação, ou seja, a palavra a ser determinada.

Essa correlação é definida por meio da maximizacão da similaridade de cossenos determinada na (Equação 25), a seguir:

$$d = \underset{i}{\operatorname{argmax}} \frac{(x_b - x_a + x_c)^\top x_i}{||x_b - x_a + x_c||} \tag{25}$$

a interpretação para a (Equação 25) é de que $x_b - x_a = x_d - x_c$, podendo ser representado pela analogia: $queen$ - $king$ = $actress - actor$; ou seja, $x_b - x_a + x_c = x_d$, sendo assim, é possível encontrar o vetor x_d, o qual maximiza o produto escalar entre os dois outros vetores de palavras. Com isso, é possível avaliar a qualidade dos vetores que representam as palavras em relação à similaridade semântica compreendida pelos humanos, capturadas nas bases citadas anteriormente.

Uma outra métrica utilizada para avaliar o resultado do *Word Embedding* é a acurácia (Equação 26), a qual utiliza o volume total de palavras que estão sendo analisadas em comparação ao número de palavras previstas corretamente, para então calcular o percentual de acerto do *Embedding* no conjunto de palavras utilizado.

$$Acurácia = \frac{Número\ de\ Palavras\ Previstas\ Corretamente}{Número\ Total\ de\ Palavras\ Analisadas} \tag{26}$$

5.1.1 Resultados para o Word2Vec

Um dos parâmetros-chaves para realizar o treinamento do *Word2Vec* é a dimensão da matriz do *embedding*, uma vez que ela estabelece o número de parâmetros que serão utilizados para determinar as relações entre as palavras. Um *embedding* com grande dimensão costuma obter melhor resultado, mas implica no fato de ser custoso. Sendo assim, se a diferença do resultado de um *embedding* de maior dimensão quando comparado a um de menor dimensão não for significativamente relevante, ou seja, com uma diferença alta entre os resultados, o presente trabalho considerou escolher o de menor dimensão para obter resultados muito próximos, porém com um custo computacional menor.

O experimento aplicado para determinar a dimensão do *Word2Vec* foi a comparação dos resultados das diferentes dimensões utilizando a análise intrínseca nas bases de dados *WordSim353* e *SimLex-999*, e a acurácia na base *Google Analogy Text*. As dimensões aplicadas foram: 150, 200, 250, 300, 350 e 400, sendo que, para essas seis dimensões treinadas, os parâmetros ajustados tinham as mesmas configurações, como pode ser visto na Tabela 3 abaixo:

Tabela 3 – Principais parâmetros do *Word2Vec* e seus valores configurados para o treinamento. A biblioteca *Gensim* foi a utilizada para realizar a tarefa.

Parâmetro no *Gensim*	**Descrição do Parâmetro**	**Valor do Parâmetro**
window	Quantidade de palavras consideradas antes e depois da palavra analisada	4
min_count	Frequência mínima da palavra na base	30
negative	Número de palavras utilizadas para comparar na amostra negativa (Seção 3.3.1)	10

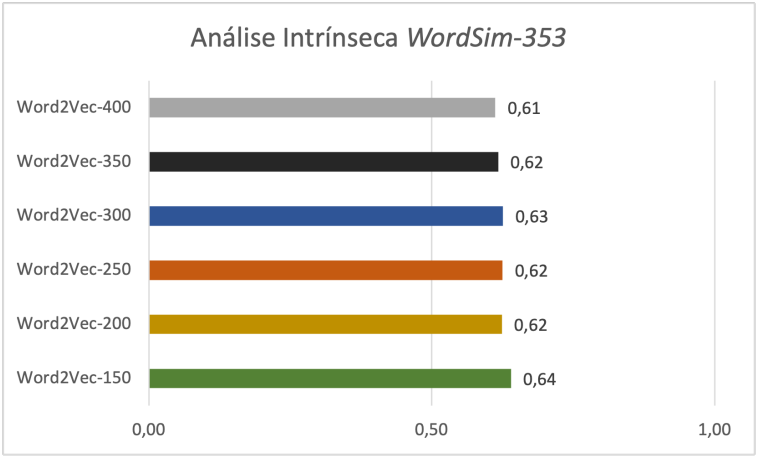
No gráfico da Figura 20 são apresentados os resultados da análise intrínseca na base de dados *WordSim-353* aplicada nas 6 dimensões treinadas, as quais apresentaram números muito próximos, com destaque para o *Word2Vec* de 150 dimensões cujo resultado gerado é o melhor. No gráfico da Figura 21 são apresentados os resultados da análise intrínseca na base de dados *SimLex-999*, onde as dimensões novamente demonstram resultados quase semelhantes, porém, o *Word2Vec* de 150 dimensões gerou um resultado muito abaixo dos demais. O mesmo ocorre no gráfico da Figura 22, no qual são apresentados os resultados da acurácia na base de dados

Google Analogy Text, o *Word2Vec* de 150 dimensões apresentou um resultado expressivo abaixo dos demais.

Esse comportamento pode ser explicado pelo tipo de avaliação que as bases de dados aplicam, o *WordSim-353* considera palavras similares por meio das relações de casualidade ou de hierarquia de palavras, como por exemplo a relação da palavra "planta"com "grama"e "alface", sendo a primeira genérica a qual implica nas palavras "grama"e "alface"serem ambas "plantas". O *SimLex-999* considera a relação puramente de similaridade entre as palavras e não apenas de associação, como por exemplo, a palavra "roupa"e "armário", no *WordSim-353* essa relação é alta, mas para o *SimLex-999* a relação entre as duas palavras é baixa, ou seja, "armário"para o *SimLex-999* é um lugar onde objetos são guardados e "roupa"é para se vestir, não há a associação de que pode guardar a "roupa"no "armário". O que pode explicar o resultado baixo do *Word2Vec* de 150 dimensões pode ser a perda das relações mais fortes, considerando a similaridade das palavras baseadas num contexto mais amplo da palavra, com menos aprofundamento.

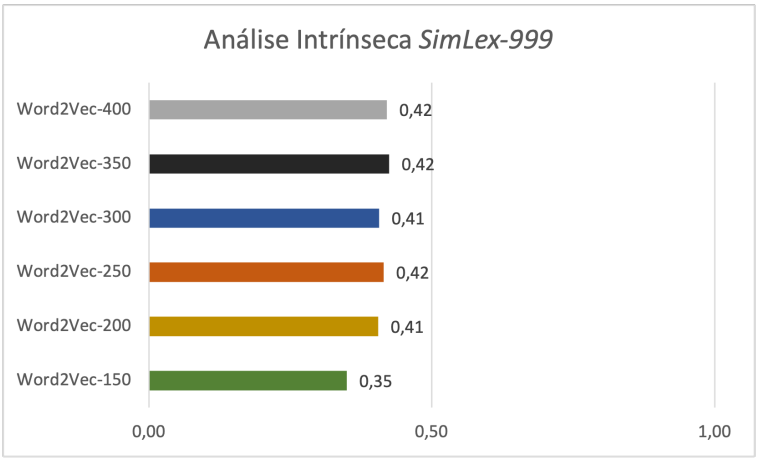
Os resultados do *Word2Vec* de 200 dimensões são muito próximos dos demais nos gráficos das Figuras 20 e 21 e superior na Figura 22. Por esse motivo, o presente trabalho irá utilizar 200 dimensões no *embedding*, esperando que o custo computacional seja reduzido quando comparado aos *embeddings* de dimensões maiores, apresentando comportamentos dos resultados semelhantes.

Figura 20 – Desempenho do *Word2Vec* em diferentes dimensões na base de dados *WordSim353*. O resultado é calculado utilizando a (Equação 25) a partir da analogia da (Equação 24).



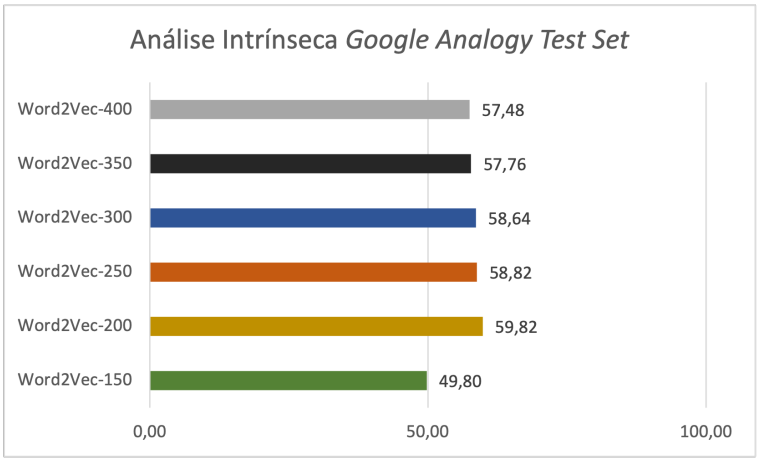
Fonte: autora.

Figura 21 – Desempenho do *Word2Vec* em diferentes dimensões na base de dados *SimLex-999*. O resultado é calculado utilizando a (Equação 25) a partir da analogia da (Equação 24).



Fonte: autora.

Figura 22 – Desempenho do *Word2Vec* em diferentes dimensões na base de dados *Google Analogy Test Set*. O resultado é calculado utilizando a acurácia (Equação 26) a partir da analogia da (Equação 24).



Fonte: autora.

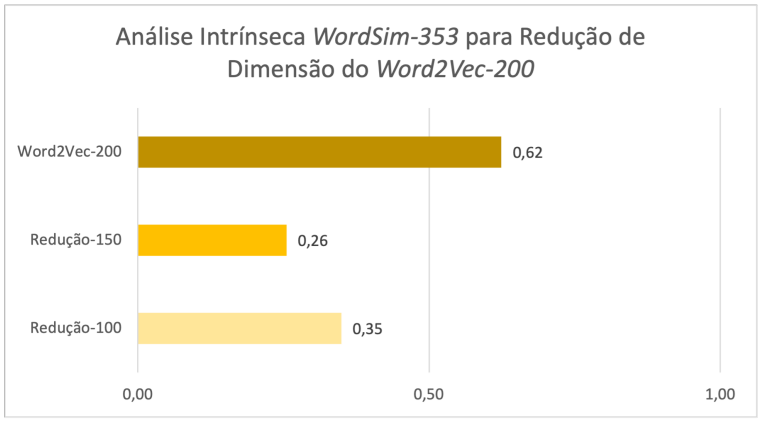
5.1.2 Resultados para a Redução de Dimensão do Word2Vec

No módulo Memória Declarativa (Seção 4.4) é aplicada a redução de dimensão (Seção 3.3.2) do *Word2Vec* gerado na Memória Perceptiva Associativa. O *Word2Vec* possui 200 dimensões, como definido na Subseção 5.1.1 anterior, sendo assim, os experimentos realizados para redução de dimensão utilizaram apenas esse *embedding*. Os experimentos foram os mesmos re-

alizados para definir a dimensão do *Word2Vec*, porém, as dimensões reduzidas testadas foram: 100 e 150.

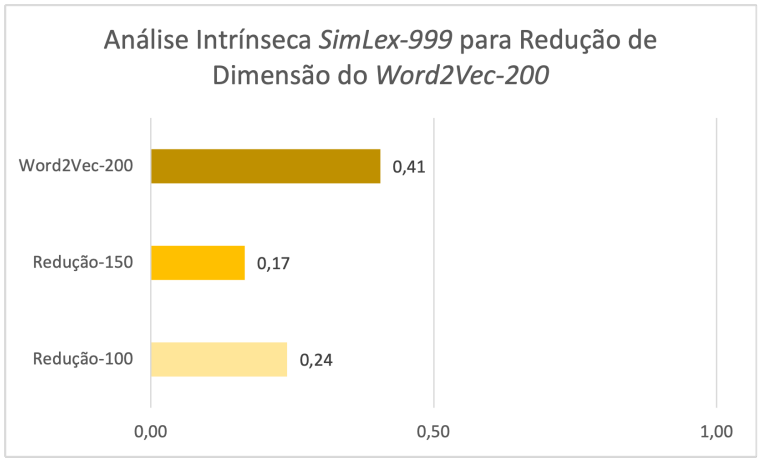
Nos gráficos das Figuras 23, 24 e 25 é apresentado o comportamento dos resultados nas bases de dados, respectivamente: *WordSim-353*, *SimLex-999* e *Google Analytics Test*. Analisando os resultados, o *Word2Vec* de 200 dimensões com redução para 100 dimensões aparece com resultados expressivamente maiores do que a redução para 150 dimensões, por esse motivo no módulo Memória Declarativa será aplicada a redução para 100 dimensões do *Word2Vec* de 200 dimensões.

Figura 23 – Desempenho do *Word2Vec* de 200 dimensões, após a aplicação de redução de dimensão, na base de dados *WordSim-353*. O resultado é calculado utilizando a (Equação 25) a partir da analogia da (Equação 24).



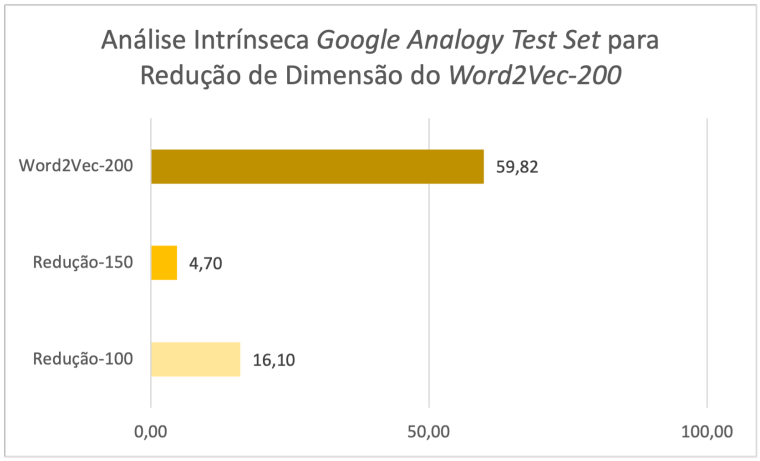
Fonte: autora.

Figura 24 – Desempenho do *Word2Vec* de 200 dimensões, após a aplicação de redução de dimensão, na base de dados *SimLex-999*. O resultado é calculado utilizando a (Equação 25) a partir da analogia da (Equação 24).



Fonte: autora.

Figura 25 – Desempenho do *Word2Vec* de 200 dimensões, após a aplicação de redução de dimensão, na base de dados *Google Analogy Test Set*. O resultado é calculado utilizando a acurácia (Equação 26) a partir da analogia da (Equação 24).

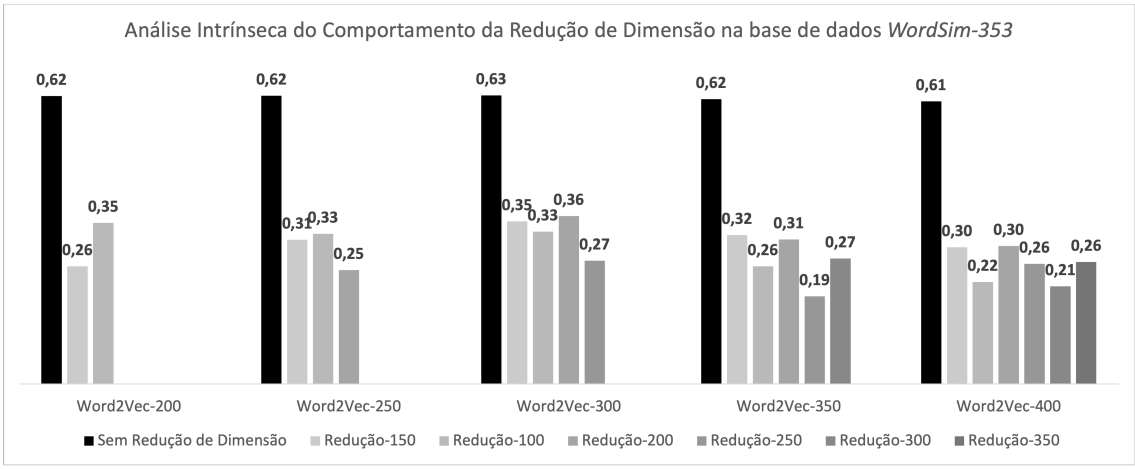


Fonte: autora.

As Figuras 26, 27 e 28 abaixo apresentam o comportamento da redução. Como pode ser observado nos gráficos de resultado da redução do *Word2Vec* de 200 dimensões, os números gerados estão abaixo dos resultados obtidos, o que influencia de forma negativa o resultado da Metodologia Proposta (Capítulo 4). Os números da redução de dimensão indicam que o *embedding* está capturando com baixa qualidade as relações das palavras, ao invés de fortalecer. Sendo assim, para entender melhor se os resultados apresentados de redução de dimensão têm a ver com o *Word2Vec* de 200 dimensões ou se está relacionado à técnica utilizada descrita na Seção 3.3.2, foi aplicada a mesma técnica para os demais *Word2Vec* de 250, 300, 350 e 400 dimensões para comparar com o desempenho do *Word2Vec* de 200 dimensões.

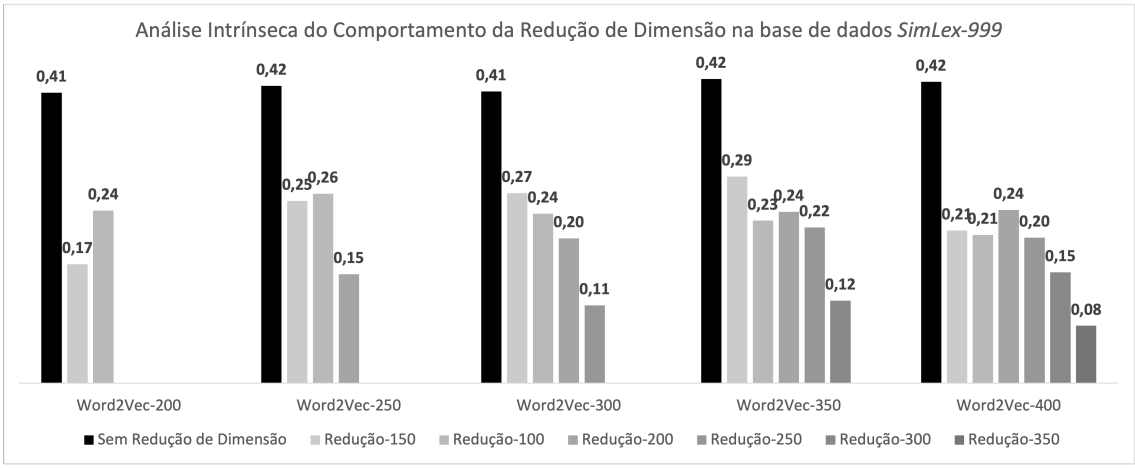
Conforme o comportamento apresentado nos gráficos, o desempenho da técnica de redução para todos os *Word2Vec* é inferior ao apresentado sem a redução. O experimento executado aplicou as reduções de acordo com o tamanho do *embedding*, como por exemplo, para o *embedding* de 200 dimensões foi aplicada a redução de 100 e 150, enquanto para o *embedding* de 400 foi aplicada a redução de 100, 150, 200, 250, 300 e 350, com o objetivo era entender o comportamento no máximo de configuração possível. Sendo assim, a técnica utilizada se mostrou ineficaz para o objetivo de sua aplicação na Memória Declarativa, a qual tem como função fortalecer as relações das palavras. O esperado da técnica seria obter um resultado próximo ou superior ao do *embedding* original. Sendo assim, serão demonstrados nas próximas seções os resultados do modelo final, o que irá determinar a importância do módulo para a metodologia como um todo.

Figura 26 – Resultado da redução aplicada no *Word2Vec* de 200, 250, 300, 350 e 400 dimensões na base de dados *WordSim-353*. As barras representam a dimensão a qual o *embedding* foi reduzido.



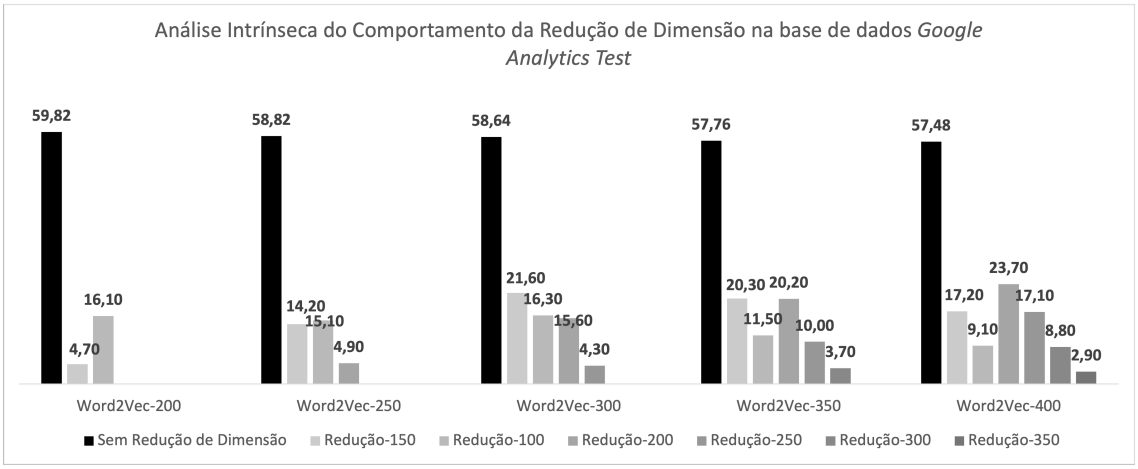
Fonte: autora.

Figura 27 – Resultado da redução aplicada no *Word2Vec* de 200, 250, 300, 350 e 400 dimensões na base de dados *SimLex-999*. As barras representam a dimensão a qual o *embedding* foi reduzido.



Fonte: autora.

Figura 28 – Resultado da redução aplicada no *Word2Vec* de 200, 250, 300, 350 e 400 dimensões na base de dados *Google Analytics Test*. As barras representam a dimensão a qual o *embedding* foi reduzido.



Fonte: autora.

5.2 EXPERIMENTOS BASEADOS NOS RESULTADOS OBTIDOS POR MEIO DA MÉTRICA ROUGE

Os experimentos apresentados nesta Seção 5.2 avaliaram a assertividade do modelo proposto na geração da Sumarização Abstrativa de Texto (Seção 3.1). Cada módulo possui uma função que contribui de alguma forma com o resultado. Por esse motivo, foram realizados quatro experimentos em que um módulo era descartado e então treinado por vez, ou seja, quatro modelos diferentes foram treinados para compreender a influência dos módulos da arquitetura do LIDA (Seção 3.4) no resultado da sumarização.

Esses experimentos foram idealizados para poder discutir os benefícios de se trabalhar com uma arquitetura modular como o LIDA, o qual possui uma teoria cognitiva pré-definida e permite o desenho de soluções através da interpretação do funcionamento de cada módulo.

Para isso, os quatro modelos foram treinados utilizando a mesma configuração e os mesmos textos, com o propósito de realizar a comparação dos modelos olhando apenas para os aspectos da retirada dos módulos. Os modelos foram treinados utilizando as bibliotecas *TensorFlow* (MARTÍN ABADI et al., 2015) e *Keras* (CHOLLET et al., 2015), e os parâmetros utilizados são aqueles mostrados na Tabela 4.

Tabela 4 – Parâmetros utilizados no treinamento dos modelos utilizando as bibliotecas *TensorFlow* e *Keras*

Parâmetro no Keras	Descrição do Parâmetro	Valor Do Parâmetro
epochs	Número de vezes que a base de dados será passada na rede	33
batch_size	Número de lotes que os dados são divididos em cada época	12
optimizer	Algoritmo de otimização utilizado no treinamento	rmsprop
loss	Função utilizada para calcular a perda	sparse_categorical_crossentropy

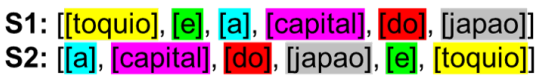
Por fim, a métrica ROUGE foi calculada para os resumos gerados pelos quatro modelos utilizando os casos da base de dados *CNN/Daily Mail* (Seção 4.1) separados como teste. O resultado da métrica foi utilizado para discutir o comportamento dos experimentos.

5.2.1 Métrica ROUGE

Para avaliação dos experimentos a seguir foram utilizadas as variações ROUGE-1, ROUGE-2 e ROUGE-L da métrica ROUGE.

A ROUGE-1 tem como objetivo avaliar em *unigram* as palavras do resumo gerado em comparação com as palavras do resumo esperado. Então, cada palavra que estiver contida igualmente nos dois resumos (sequência) é definida como uma palavra gerada pelo ROUGE-1. A Figura 29 exemplifica a separação em unigram, sendo que, cada cor representa uma palavra nas duas sequências avaliadas.

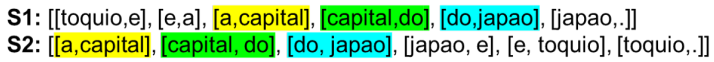
Figura 29 – Exemplo de como é feita a contagem de palavras que co-ocorrem na métrica ROUGE-1. Cada cor representa um *unigram* que co-ocorre nas duas sequências avaliadas.



Fonte: autora.

A ROUGE-2 tem como objetivo avaliar em *bigram* as palavras do resumo gerado em comparação com as palavras do resumo esperado. Semelhante ao que ocorre na ROUGE-1, a ROUGE-2 contabiliza o número de *bigrams* que co-ocorrem na sequência gerada pelo modelo e na sequência do resumo esperada. A Figura 30 exemplifica a separação em *bigram*, sendo que, cada cor representa uma *bigram* correspondente nas duas sequências avaliadas.

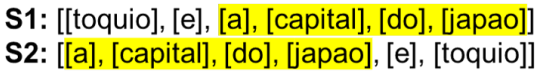
Figura 30 – Exemplo de como é feita a contagem de palavras que co-ocorrem na métrica ROUGE-2. Cada cor representa um *bigram* que co-ocorre nas duas sequências avaliadas.



Fonte: autora.

A ROUGE-L tem como objetivo avaliar a subsequência mais longa em comum entre as duas sequências de resumo geradas e resumo esperado. Para essa métrica é retornado o número de elementos que ocorrem nessa subsequência. A Figura 31 exemplifica como é considerada a subsequência mais longa em comum.

Figura 31 – Exemplo de como é feita a contagem de palavras que co-ocorrem na métrica ROUGE-L. As palavras que estão destacadas representam a subsequência mais longa em comum entre as duas sequências avaliadas.



Fonte: autora.

A métrica ROUGE utiliza ainda três métodos para calcular a qualidade das sumarizações para cada umas das três métricas selecionadas (ROUGE-1, ROUGE-2 e ROUGE-L), são eles: precisão, *recall* e *F1-Score*.

A Precisão (P) é calculada utilizando o número de palavras que co-ocorrem (sobrepostas) entre as duas sequências sobre o número de palavras do resumo gerado, como mostrado na (Equação 27).

$$P = \frac{\text{Nº de palavras sobrepostas}}{\text{Nº de palavras do resumo gerado}} \tag{27}$$

O *Recall* (R) é calculado utilizando o número de palavras que co-ocorrem (sobrepostas) entre as duas sequências sobre o número de palavras do resumo esperado (gabarito), como mostrado na (Equação 28).

$$R = \frac{\text{Nº de palavras sobrepostas}}{\text{Nº de palavras do resumo gabarito}} \tag{28}$$

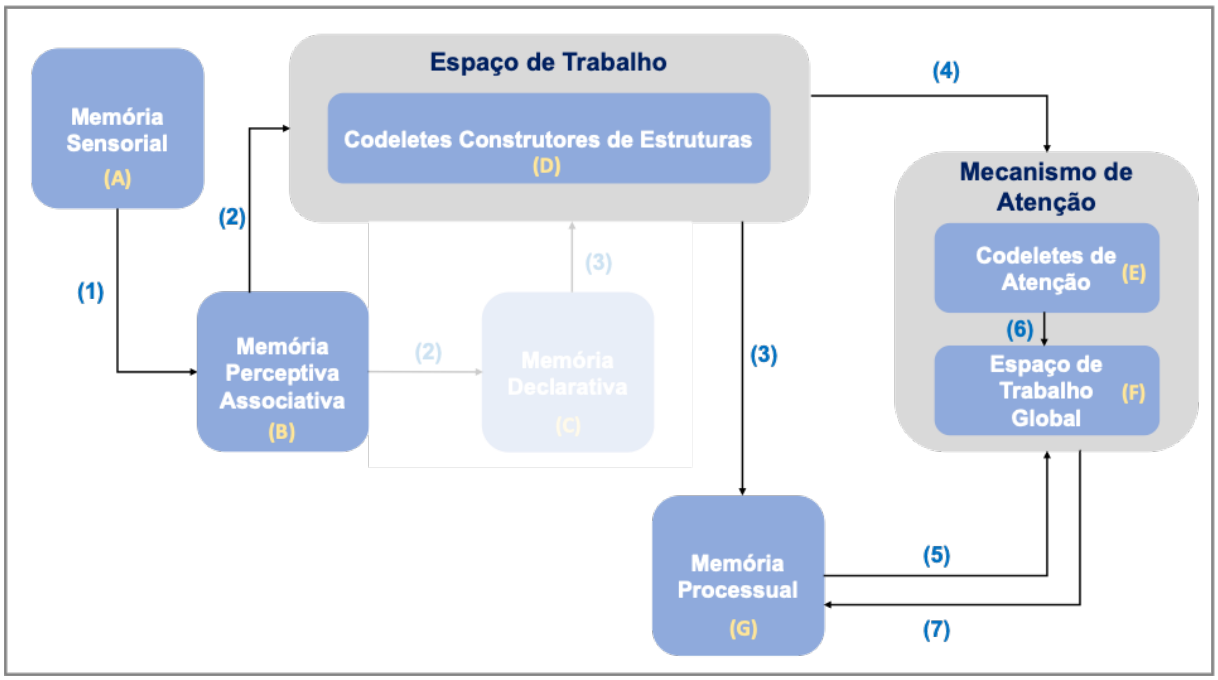
O *F1-Score* (F1) é calculado levando em consideração a precisão e o *recall* obtidos. A (Equação 29) demonstra o cálculo.

$$F1 = 2 * \frac{R * P}{R + P} \tag{29}$$

5.2.2 Resultados Descartando o Módulo Memória Declarativa

Com o propósito de avaliar o comportamento do modelo proposto após a retirada do módulo Memória Declarativa, o experimento a seguir realizou o treino do modelo utilizando os módulos apresentados na Figura 32. A Memória Declarativa tem como função realizar a redução de dimensão da matriz de *embedding* gerada pela Memória Perceptiva Associativa (Figura 32, Bloco B), com a finalidade de intensificar as relações entre as palavras utilizando a técnica descrita na Seção 3.3.2 e reforçada na Seção 4.4. Sendo assim, com a retirada do módulo a matriz de *embedding* gerada na Memória Perceptiva Associativa é enviada diretamente para o módulo Codeletes Construtores de Estruturas, sem redução de dimensão.

Figura 32 – Modelo proposto de sumarização de texto baseado no modelo teórico cognitivo LIDA após a retirada do módulo Memória Declarativa. Em amarelo, são apresentadas as letras que se referem aos módulos do modelo, e em azul, são apresentados os números que se referem às conexões entre os módulos.

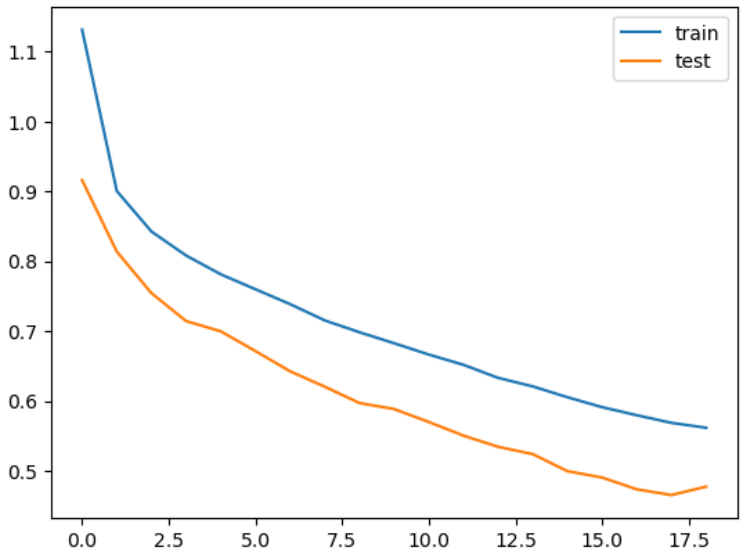


Fonte: autora.

No gráfico apresentado na Figura 33 é possível observar o comportamento dos valores de perda ao longo do treinamento do modelo. O modelo foi treinado utilizando as configurações detalhadas na Tabela 4, e ainda, foi aplicado *Early Stopping* considerando o aumento da perda para duas épocas seguidas. O treinamento desse modelo teve tempo de execução de dois dias, duas horas e cinco minutos, sendo que executou dezoito épocas. Nesse gráfico, a curva de treino possui perda menor que a curva de teste, indicando que o conjunto de dados de teste

é mais fácil de prever do que o conjunto de dados de treinamento. Quando comparado ao gráfico do modelo treinado sem *Word2Vec* (Memória Perceptiva Associativa), as curvas estão mais próximas, indicando uma possível melhora no treinamento em relacão ao modelo treinado na seção anterior. Mesmo assim, o ajuste do modelo sem redução não apresenta uma curva de aprendizado ideal, elas estão distantes, o que pode causar erros significativos na geração da sumarização.

Figura 33 – Gráfico da curva de aprendizado modelo sem a Memória Declarativa utilizando a perda, calculada por meio da Entropia Cruzada Categórica. O eixo x descreve o número de épocas (*epochs*) que o modelo executou e o eixo y o valor da perda. A curva azul representa a perda no conjunto de dados de teste e a curva laranja a perda no conjunto de dados de treino.

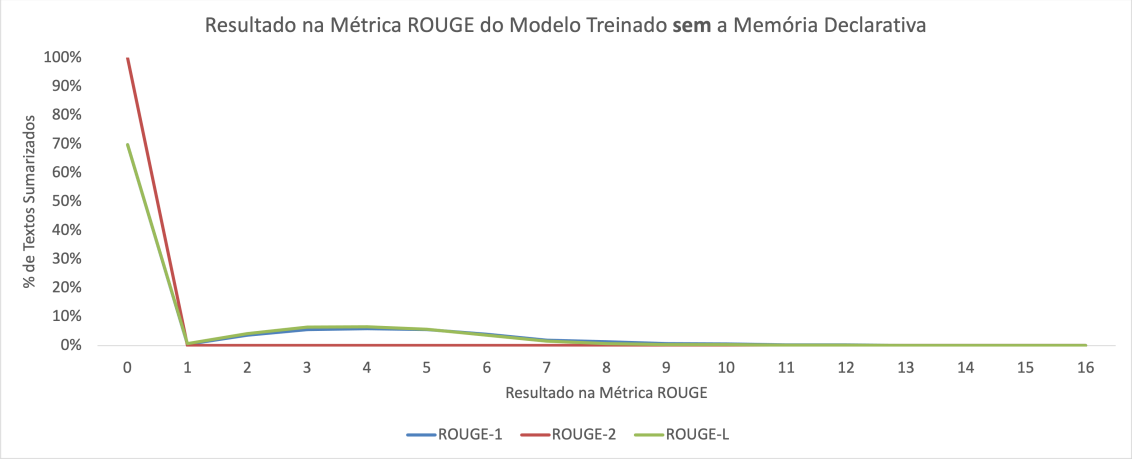


Fonte: autora.

Para avaliar os resultados da sumarização gerada pelo modelo foi utilizado o *F1 score* da métrica ROUGE, apresentada na Seção 5.2.1. No gráfico da Figura 34, as curvas de concentração dos *scores* gerados pelas métricas também possuem alta concentração no ponto inicial 0. Na variação ROUGE-1, 70% das sumarizações realizadas tiveram pontuação 0 e 1, resultado novamente baixo, dado que a ROUGE-1 avalia a *unigram* sobreposta entre o resumo esperado e o gerado. A pontuação máxima alcançada para uma sumarização foi de 16 pontos. Na Tabela 2 são apresentados os resultados dos trabalhos relacionados para as métricas utilizadas nesse experimento. A pontuação da ROUGE-1 média dos 11 trabalhos citados que utilizaram a base de dados *CNN/Daily Mail* foi de 39,36. Na variação ROUGE-2, o resultado foi igual ao do modelo treinado sem a Memória Perceptiva Associativa, 100% das sumarizações não tiveram *bigram* sobrepostas entre o resumo esperado e gerado. E para a variação ROUGE-L, 71% dos

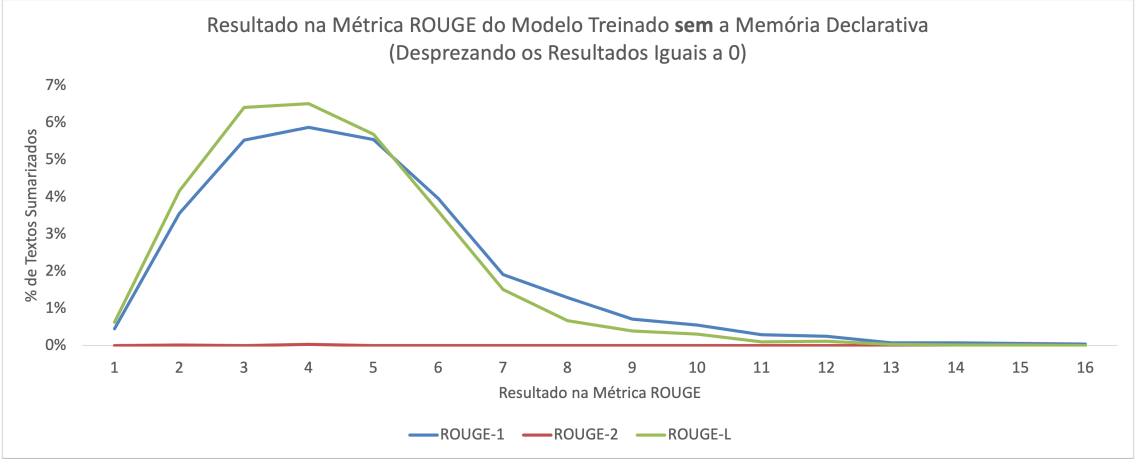
resultados se concentraram nas pontuações 0 e 1, tendo como pontuação máxima atingida por uma sumarização igual a 8, a média de 10 dos 11 trabalhos citados é de 33,5 pontos. O gráfico da Figura 35 permite observar o comportamento das pontuações diferentes de 0, o eixo x determina o percentual que é distribuído nas faixas maiores que 0, de menor concentração, o que permite observar que as curvas da ROUGE-1 e ROUGE-L possuem comportamento semelhante, enquanto a ROUGE-2 se estabiliza no 0% por não ter resultados acima de 0.

Figura 34 – Gráfico do comportamento das sumarizações realizadas pelo modelo sem Memória Declarativa na métrica ROUGE. Cada curva é referente a uma variação da métrica sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE.



Fonte: autora.

Figura 35 – Gráfico do comportamento das sumarizações realizadas pelo modelo sem Memória Declarativa na métrica ROUGE. Cada curva é referente a uma variação da métrica sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE. Neste gráfico é descartada a pontuação ROUGE igual a 0 para observar o comportamento das curvas de percentual mais baixo.

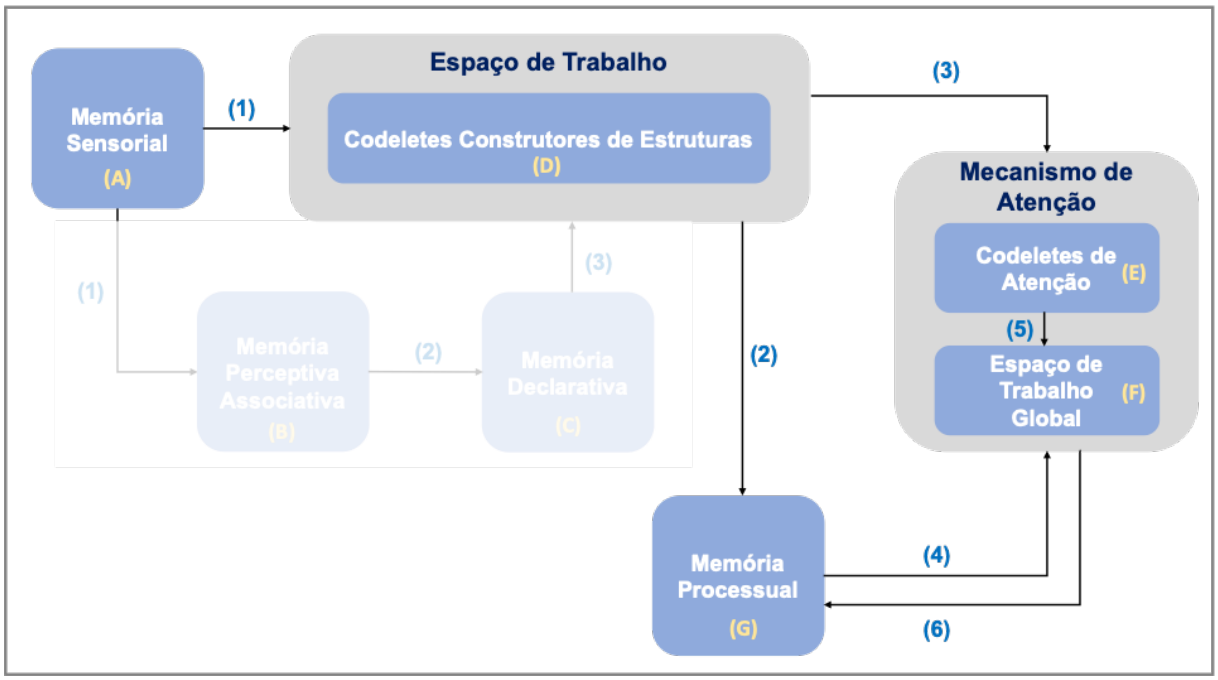


Fonte: autora.

5.2.3 Resultados Descartando os Módulos Memória Perceptiva Associativa e Memória Declarativa

Para analisar o desempenho do modelo proposto após a retirada do módulo Memória Perceptiva Associativa, o experimento a seguir realizou o treino do modelo utilizando os módulos apresentados na Figura 36. A Memória Perceptiva Associativa é responsável por gerar a matriz de *embedding*, a qual é criada utilizando o *Word2Vec*. O treinamento do *Word2Vec* é realizado a parte e, portanto, com a retirada do módulo que o executa, o treinamento é realizado junto aos demais módulos: Codeletes Construtores de Estruturas (Figura 36, D) e Memória Processual (Figura 36, G). Como pode ser observado na Figura 36, o módulo Memória Declarativa também é descartado, pois seu funcionamento depende da matriz de *embedding* vinda da Memória Perceptiva Associativa, seu desenvolvimento é detalhado na Seção 4.4.

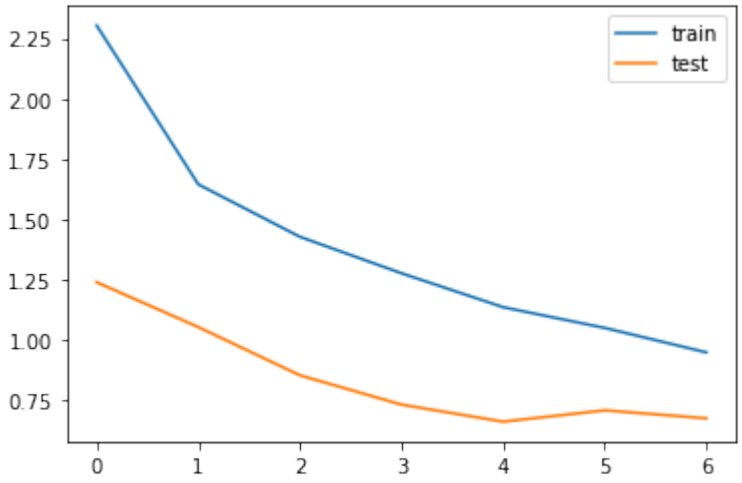
Figura 36 – Modelo proposto de sumarização de texto baseado no modelo teórico cognitivo LIDA após a retirada dos módulos: Memória Perceptiva Associativa e Memória Declarativa. Em amarelo, são apresentadas as letras que se referem aos módulos do modelo, e em azul, são apresentados os números que se referem às conexões entre os módulos.



Fonte: autora.

No gráfico apresentado na Figura 37 é possível observar o comportamento dos valores de perda ao longo do treinamento do modelo. O modelo foi treinado utilizando as configurações detalhadas na Tabela 4, e ainda, foi aplicado *Early Stopping* considerando o aumento da perda para duas épocas seguidas. O treinamento desse modelo teve tempo de execução de um dia, sete horas e trinta e três minutos, sendo que executou seis épocas. Nesse gráfico, a curva de treino possui perda menor que a curva de teste, indicando que o conjunto de dados de teste é mais fácil de prever do que o conjunto de dados de treinamento. Isso pode ocorrer quando os dados de teste são pequenos, mas representam bem o conjunto de dados de treino, o que leva o modelo a ter um desempenho muito bom nesses poucos exemplos. Sendo assim, o modelo não obteve um bom ajuste, o que pode causar altos erros na geração da sumarização.

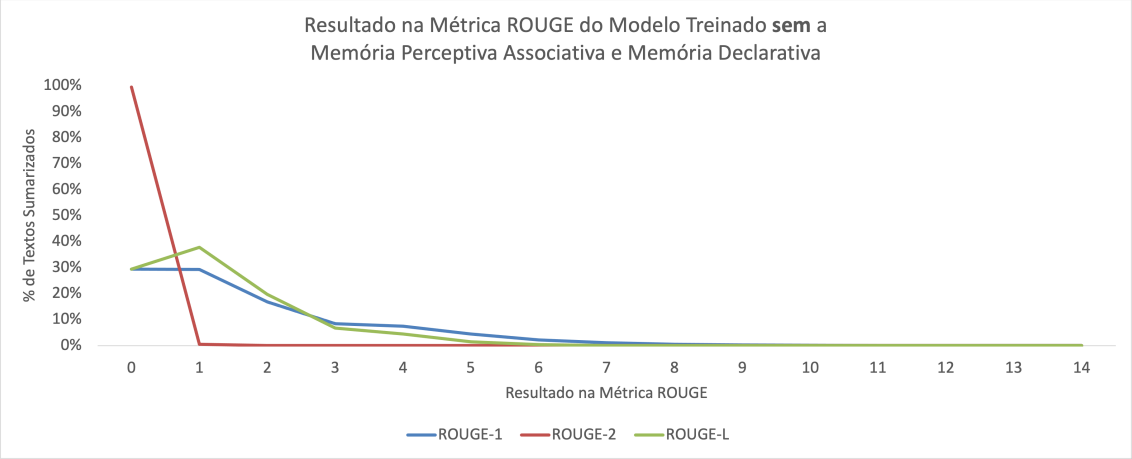
Figura 37 – Gráfico da curva de aprendizado do modelo sem a Memória Perceptiva Associativa e Memória Declarativa utilizando a perda, calculada por meio da Entropia Cruzada Categórica. O eixo x descreve o número de épocas (*epochs*) que o modelo executou e o eixo y o valor da perda. A curva azul representa a perda no conjunto de dados de teste e a curva laranja a perda no conjunto de dados de treino.



Fonte: autora.

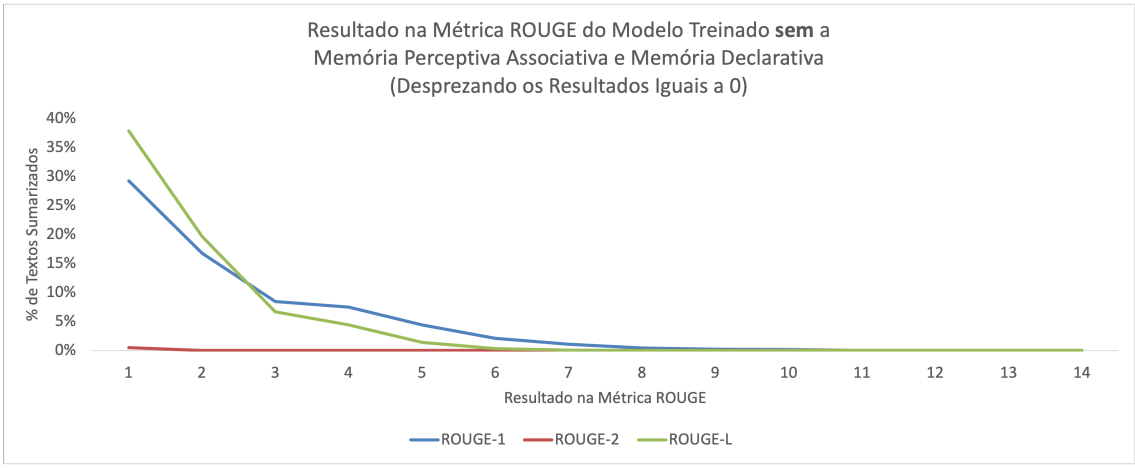
Para avaliar os resultados da sumarização gerada pelo modelo foi utilizado o *F1 score* da métrica ROUGE, apresentada na Seção 5.2.1. No gráfico da Figura 38 as curvas de concentração dos *scores* gerados pelas métricas possuem alta concentração em 0. Na variação ROUGE-1, 59% das sumarizações realizadas tiveram pontuação 0 e 1, resultado muito baixo, dado que a ROUGE-1 avalia a *unigram* sobreposta entre o resumo esperado e o gerado. A pontuação máxima alcançada para uma sumarização foi de 14 pontos. Na Tabela 2 (Capítulo 2) são apresentados os resultados dos trabalhos relacionados para as métricas utilizadas nesse experimento. A pontuação da ROUGE-1 média dos 11 trabalhos citados que utilizaram a base de dados *CNN/Daily Mail* foi de 39,36. Na variação ROUGE-2 100% das sumarizações não tiveram *bigram* sobrepostas entre o resumo esperado e gerado. E para a variação ROUGE-L, muito semelhante ao comportamento da ROUGE-1, 67% dos resultados se concentraram nas pontuações 0 e 1, tendo como pontuação máxima atingida por uma sumarização igual a 5, a média de 10 dos 11 trabalhos citados é de 33,5 pontos. O gráfico da Figura 39 permite observar o comportamento das pontuações diferentes de 0, o eixo x determina o percentual que é distribuído nas faixas maiores que 0, de menor concentração.

Figura 38 – Gráfico do comportamento das sumarizações realizadas pelo modelo sem Memória Perceptiva Associativa e Memória Declarativa na métrica ROUGE. Cada curva é referente a uma variação das métricas, sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE.



Fonte: autora.

Figura 39 – Gráfico do comportamento das sumarizações realizadas pelo modelo sem Memória Perceptiva Associativa e Memória Declarativa na métrica ROUGE. Cada curva é referente a uma variação das métricas, sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE. Neste gráfico, é descartada a pontuação ROUGE igual a 0 para observar o comportamento das curvas de percentual mais baixo.

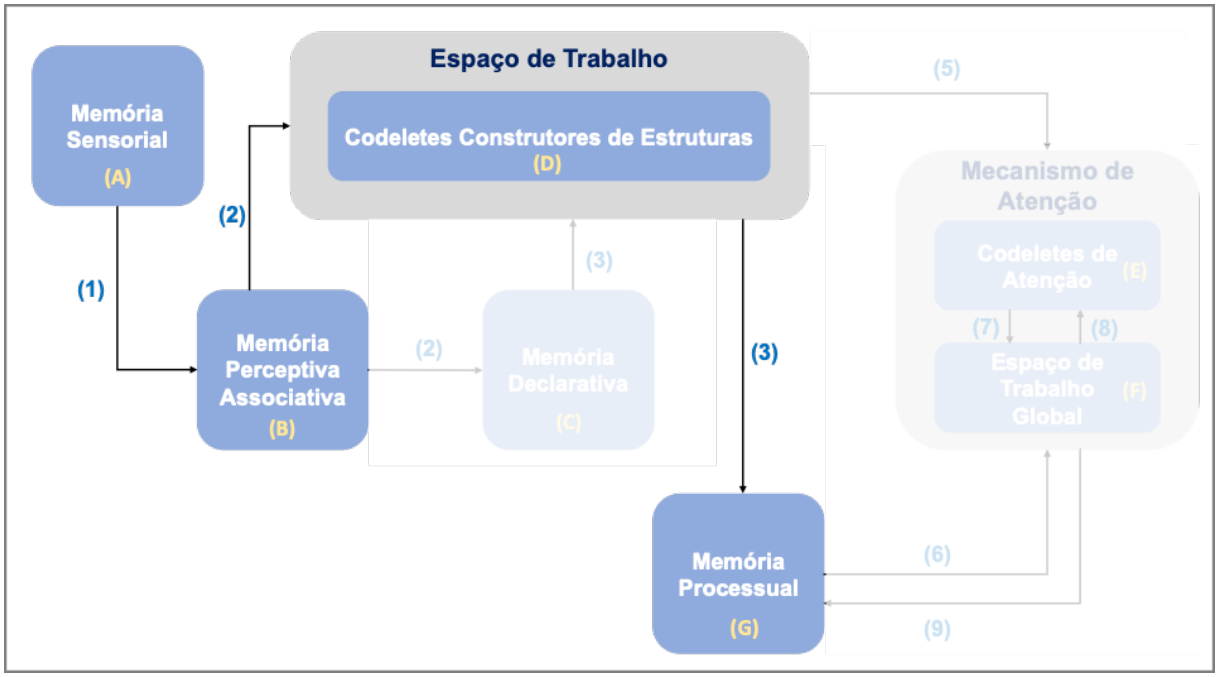


Fonte: autora.

5.2.4 Resultados Descartando os Módulos Codeletes de Atenção e Espaço de Trabalho Global

Dando sequência aos experimentos que avaliam o comportamento dos módulos do modelo proposto, o experimento a seguir realizou o treino do modelo descartando os Codeletes de Atenção e o Espaço de Trabalho Global. Na Figura 40 são apresentados os módulos utilizados no treino. Os Codeletes de Atenção e o Espaço de Trabalho Global fazem parte do Mecanismo de Atenção e respectivamente têm como função: determinar a pontuação das palavras e determinar a palavra com maior pontuação (vencedora). Com a retirada dos dois módulos, o modelo passa a utilizar apenas o resultado da última camada do *encoder* situado nos Codeletes Construtores de Estruturas (Figura 40, D), o qual é enviado diretamente para o *decoder* que está localizado na Memória Processual (Figura 40, G); ou seja, o modelo irá considerar sempre os pesos atribuídos na última camada, o que pode excluir sentenças que possuem maior contribuição para geração do resumo.

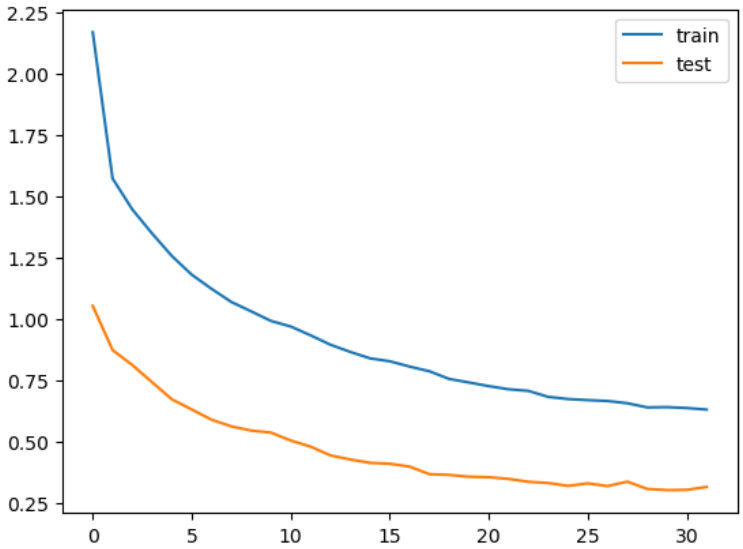
Figura 40 – Modelo proposto de sumarização de texto baseado no modelo teórico cognitivo LIDA após a retirada dos módulos: Codeletes de Atenção e Espaço de Trabalho Global. Em amarelo, são apresentadas as letras que se referem aos módulos do modelo, e em azul, são apresentados os números que se referem às conexões entre os módulos.



Fonte: autora.

No gráfico apresentado na Figura 41 é possível observar o comportamento dos valores de perda ao longo do treinamento do modelo. O modelo foi treinado utilizando as configurações detalhadas na Tabela 4, e ainda, foi aplicado *Early Stopping* considerando o aumento da perda para duas épocas seguidas, assim como foi aplicado nos modelos treinados nas seções anteriores. O treinamento desse modelo teve tempo de execução de quatro dias, uma hora e trinta e seis minutos, sendo que executou trinta e duas épocas. No gráfico da Figura 41 a curva de treino possui perda menor que a curva de teste, indicando que o conjunto de dados de teste é mais fácil de prever do que o conjunto de dados de treinamento. Quando comparada às curvas dos modelos treinados e apresentados nas seções anteriores, percebe-se que houve uma queda constante no valor da perda durante o treino, chegando ao valor próximo de 0. Porém, as curvas de treino e teste ao longo das 32 épocas continuam distantes, demonstrando que o modelo, assim como os anteriores, também não apresentou uma curva de aprendizado ideal para seu ajuste.

Figura 41 – Gráfico da curva de aprendizado do modelo sem os Codeletes de Atenção e Espaço de Trabalho Global utilizando a perda, calculada por meio da Entropia Cruzada Categórica. O eixo x descreve o número de épocas (*epochs*) que o modelo executou e o eixo y o valor da perda. A curva azul representa a perda no conjunto de dados de teste e a curva laranja a perda no conjunto de dados de treino.

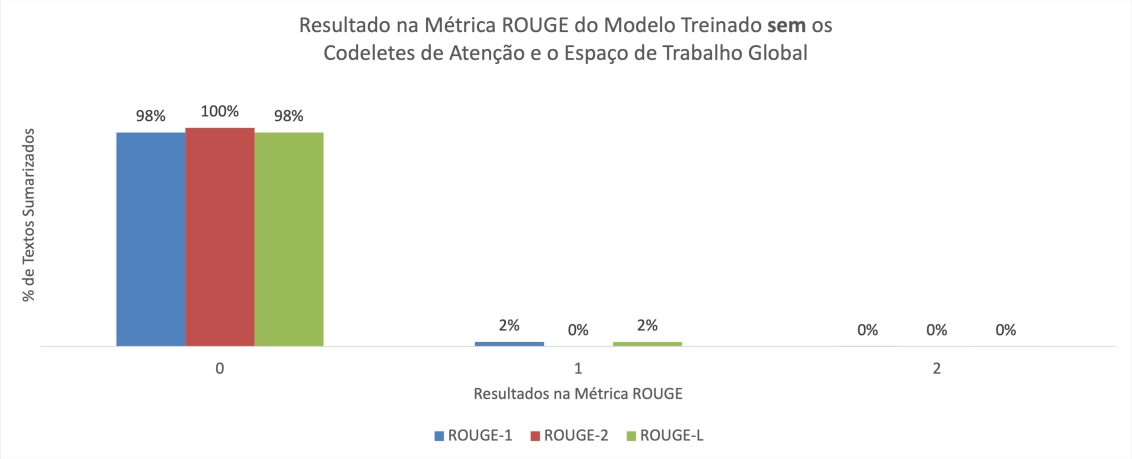


Fonte: autora.

Para avaliar os resultados da sumarização gerada pelo modelo foi utilizado o *F1 score* da métrica ROUGE, apresentada na Seção 5.2.1. No gráfico da Figura 42 as curvas de concentração dos *scores* gerados pelas métricas se concentraram na pontuação 0. Essa alta concentração demonstra que o modelo aleatorizou a geração das palavras para compor o resumo, em razão da retirada dos módulos Codeletes de Atenção e Espaço de Trabalho Global, a Memória Processual

recebe dos Codeletes Construtores de Estruturas um vetor de tamanho fixo do último estado da camada oculta do *encoder*. Devido a isso, o vetor não guarda toda a informação do contexto necessário para resumir o texto. Esse problema normalmente é solucionado com o Mecanismo de Atenção e pode ser incrementado com outras técnicas como *Pointer-Generator* apresentado em (SEE; LIU; MANNING, 2017). Sendo assim, era esperado que o modelo sem os dois módulos tivesse resultados piores que os demais modelos treinados.

Figura 42 – Gráfico do comportamento das sumarizações realizadas pelo modelo sem os Codeletes de Atenção e Espaço de Trabalho Global na métrica ROUGE. Cada barra é referente a uma variação das métricas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE.



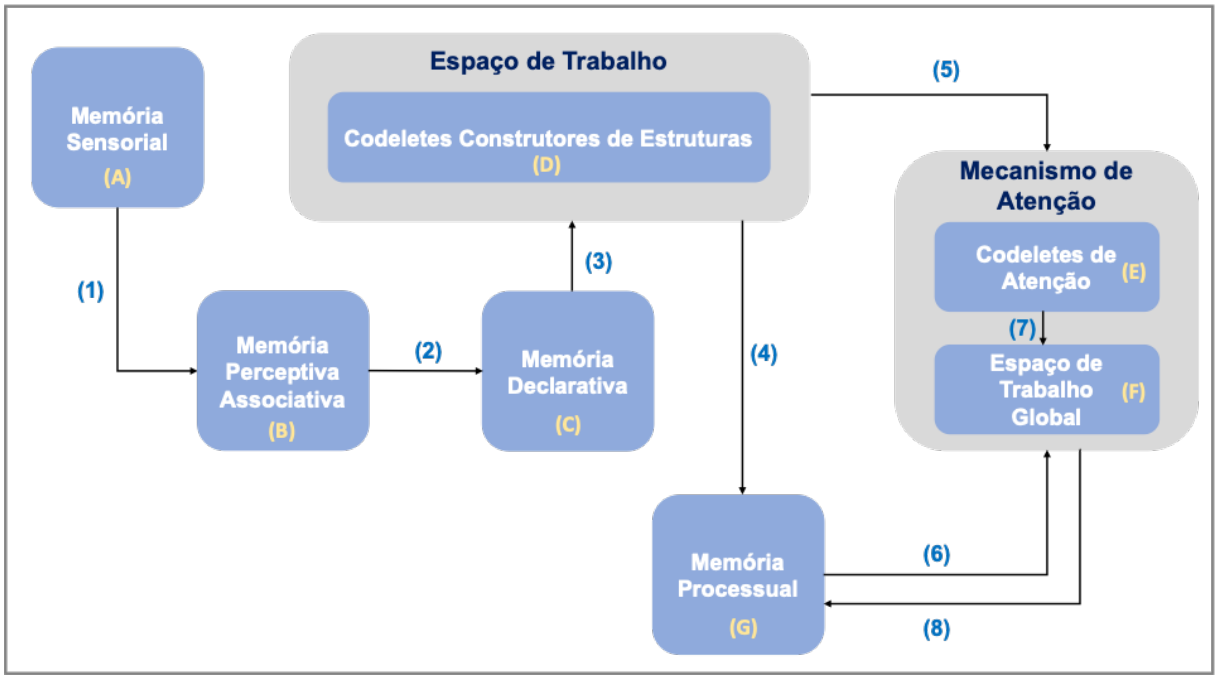
Fonte: autora.

5.2.5 Resultado do Modelo Proposto Utilizando Todos os Módulos

O experimento a seguir foi realizado utilizando todos os módulos do modelo proposto, apresentado na Figura 43. A Memória Sensorial (Figura 43, A) é responsável pelo pré-processamento do texto, realizando o tratamento do texto e o transformando em *tokens*, os quais são enviados para a Memória Perceptiva Associativa (Figura 43, B). Essa memória então utiliza os *tokens* para criar uma matriz de *embeddings* que contém as relações entre as palavras, e as envia para o próximo módulo. A Memória Declarativa (Figura 43, C) recebe a matriz de *embeddings* e aplica uma técnica de redução de dimensão com o objetivo de intensificar a singularidade de cada palavra tornando o *embedding* mais discriminativo. Os Codeletes Construtores de Estruturas (Figura 43, D) irá utilizar a matriz vinda da Memória Declarativa como entrada da rede neural codificadora que irá gerar o vetor de contexto do texto, utilizando a última camada da

rede, a qual será enviada para a Memória Processual (Figura 43, G) e para o Mecanismo de Atenção. A Memória Processual irá utilizar o vetor gerado pelo codificador com o contexto do texto para determinar qual deverá ser a palavra gerada em cada passo para compor resumo. Para isso, a cada instante, os Codeletes de Atenção (Figura 43, E) e o Espaço de Trabalho Global (Figura 43, F), respectivamente, calculam a pontuação de cada estado da última camada da rede gerada nos Codeletes Construtores de Estruturas e os envia para o Espaço de Trabalho Global, que irá determinar qual o vencedor, zerando os estados que possuem menor peso. Sendo assim, o decodificador da Memória Processual irá utilizar o resultado vindo do Espaço de Trabalho Global para determinar qual palavra deve ser gerada para compor o resumo, passo a passo.

Figura 43 – Modelo proposto de sumarização de texto baseado no modelo teórico cognitivo LIDA. Em amarelo, são apresentadas as letras que se referem aos módulos do modelo, e em azul, são apresentados os números que se referem às conexões entre os módulos.

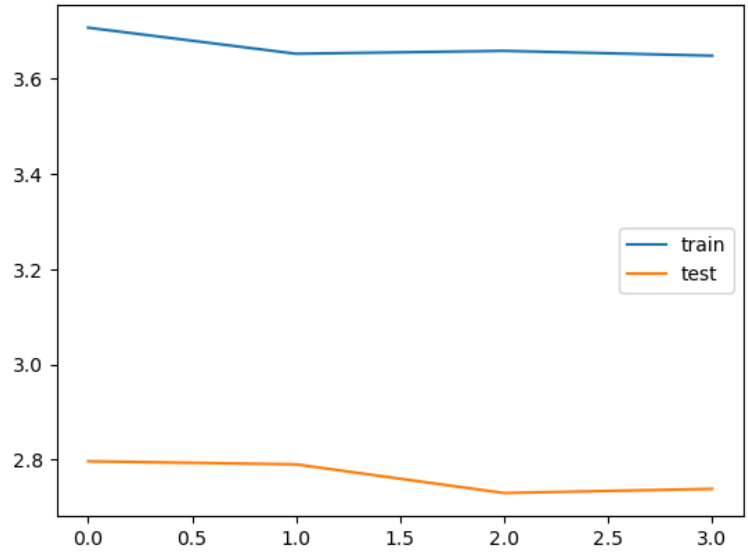


Fonte: autora.

No gráfico apresentado na Figura 44 é possível observar o comportamento dos valores de perda ao longo do treinamento do modelo. O modelo foi treinado utilizando as configurações detalhadas na Tabela 4, e ainda, foi aplicado *Early Stopping* considerando o aumento da perda para 2 épocas seguidas, assim como foi aplicado nos modelos treinados nas seções anteriores. O treinamento desse modelo teve tempo de execução de 11 horas e 2 minutos, sendo que executou 3 épocas. No gráfico da Figura 44 a curva de treino e teste estão distantes e muito estáveis, ou seja, ocorreu *underfitting*, pois o modelo não conseguiu aprender com o conjunto de dados

de treino, observado pela estabilidade da perda a qual não diminuiu ao longo das épocas, demonstrando que o modelo não teve capacidade adequada para compreender a complexidade ou o comportamento do conjunto de dados de treino.

Figura 44 – Gráfico da curva de aprendizado do modelo proposto utilizando a perda, calculada por meio da Entropia Cruzada Categórica. O eixo x descreve o número de épocas (*epochs*) que o modelo executou e o eixo y o valor da perda. A curva azul representa a perda no conjunto de dados de teste e a curva laranja a perda no conjunto de dados de treino.

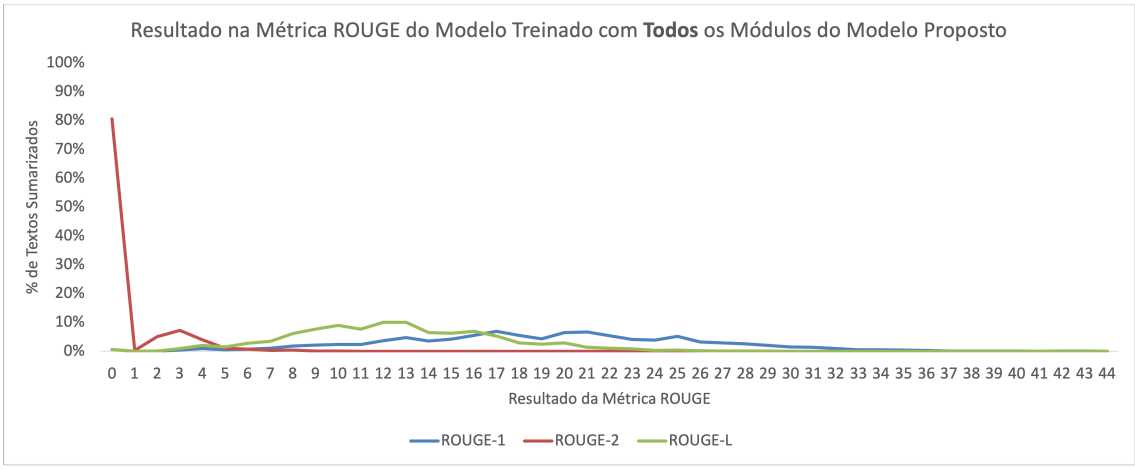


Fonte: autora.

Na Seção 5.1.2 foram apresentados os experimentos feitos exclusivamente com a matriz de *embedding* reduzida, que demonstrou um desempenho ruim nos conjuntos de dados específicos para testar a qualidade de *word embedding*. Sendo assim, é possível afirmar que o módulo Memória Declarativa, não contribui com os resultados do modelo proposto para realizar sumarização de texto, pois quando retirado esse módulo, o desempenho do modelo melhora, portanto, aos demais experimentos realizados nesse trabalho o módulo Memória Declarativa foi retirado para que o desempenho do treino não fosse afetado.

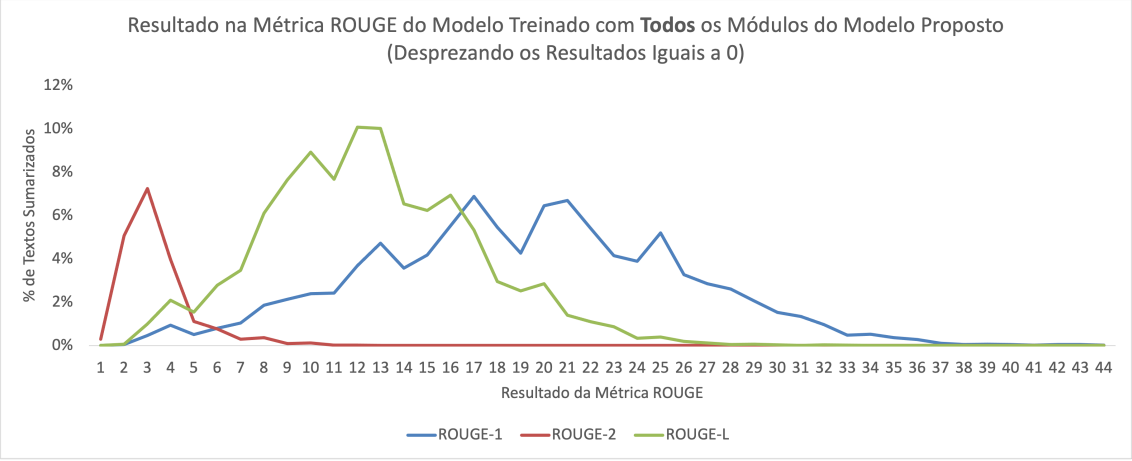
O modelo proposto gerou para todos os textos do conjunto de teste a mesma sumarização. A frase gerada para todos os textos foi a seguinte: "*the the the the num to num to in a in a*", ou seja, palavras muito comuns nos resumos esperados e que serão encontradas de maneira frequente. Nas Figuras 45 e 46 os resultados na métrica ROUGE parecem melhores quando comparados aos demais modelos treinados, porém, percebe-se uma limitação da métrica. A ROUGE não avalia de fato a coerência e coesão do texto e pode gerar falsos positivos por apenas comparar palavras que existem no resumo esperado com o resumo gerado.

Figura 45 – Gráfico do comportamento das sumarizações realizadas pelo modelo proposto na métrica ROUGE. Cada curva é referente a uma variação das métricas, sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE.



Fonte: autora.

Figura 46 – Gráfico do comportamento das sumarizações realizadas pelo modelo proposto na métrica ROUGE. Cada curva é referente a uma variação das métricas, sendo elas: ROUGE-1, ROUGE-2 e ROUGE-L. O eixo x representa o percentual de textos que foram sumarizados e o eixo y reflete a pontuação da ROUGE. Neste gráfico é descartada a pontuação ROUGE igual a 0 para observar o comportamento das curvas de percentual mais baixo.



Fonte: autora.

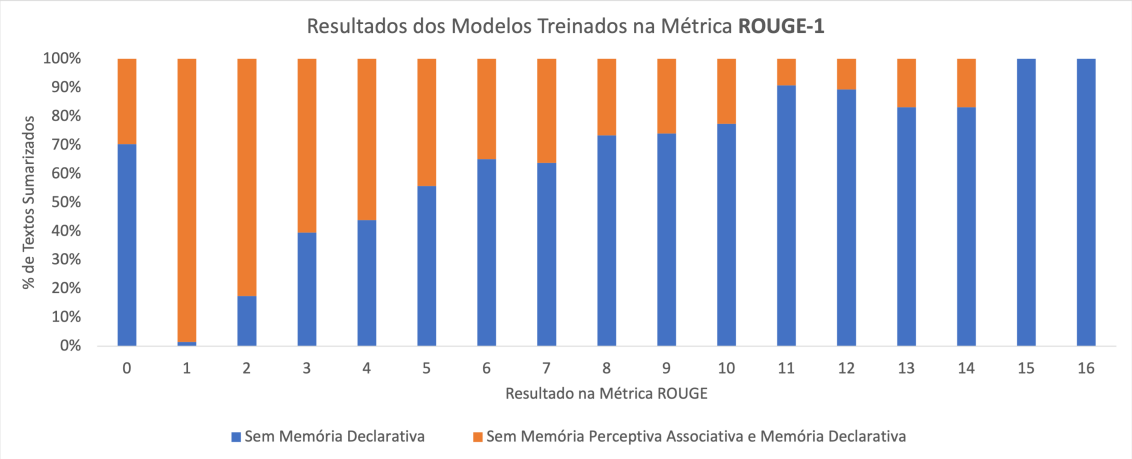
5.2.6 Análise Comparativa dos Resultados Obtidos pelos Modelos Treinados

Os experimentos executados neste trabalho demonstraram que a técnica de redução de dimensão aplicada na Memória Declarativa não contribui com o resultado da sumarização. Por

esse motivo, o modelo final proposto irá desconsiderar esse módulo. Uma outra análise realizada foi a contribuição dos Codeletes de Atenção e Espaço de Trabalho à sumarização, os módulos se mostraram essenciais para a determinação das sentenças, tendo em vista os resultados obtidos com sua retirada que demonstrou aleatoriedade do modelo na escolha das palavras que compõem o resumo. Por outro lado, os resultados obtidos por meio dos treinos dos modelos com e sem a Memória Perceptiva Associativa apresentaram pontuações mais altas.

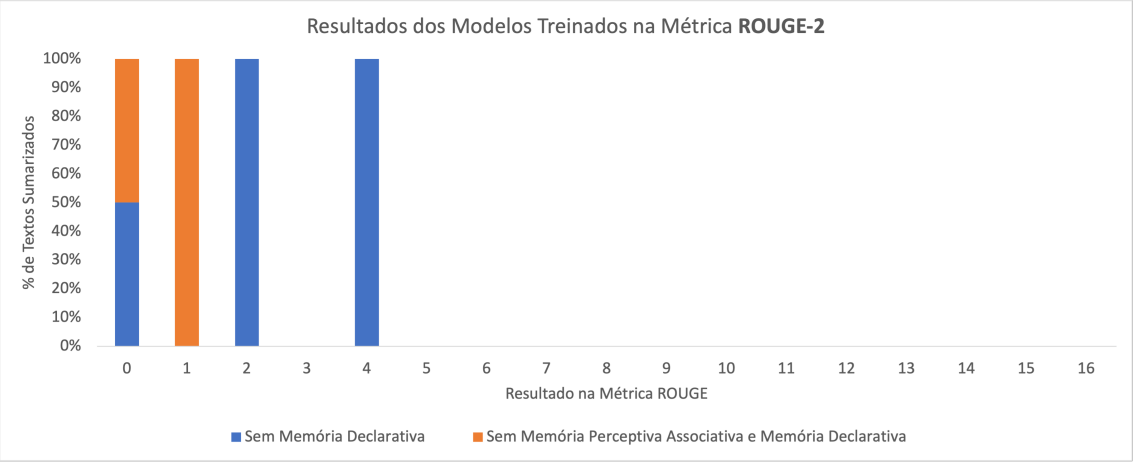
Os gráficos das Figuras 47, 48 e 49 apresentam, respectivamente, os resultados na ROUGE-1, ROUGE-2 e ROUGE-L dos modelos treinados com e sem a Memória Perceptiva Associativa. Quando comparadas as distribuições, nota-se uma maior concentração nas pontuações mais altas pelo modelo treinado apenas com a retirada da Memória Declarativa, apesar de uma alta concentração na pontuação 0, o modelo possui maior potencial de alcançar melhores pontuações quando o módulo Memória Perceptiva Associativa é utilizado.

Figura 47 – Gráfico com a distribuição das pontuações obtidas na ROUGE-1 pelos modelos treinados: sem Memória Declarativa (em azul) e sem Memória Perceptiva Associativa e Memória Declarativa (em laranja).



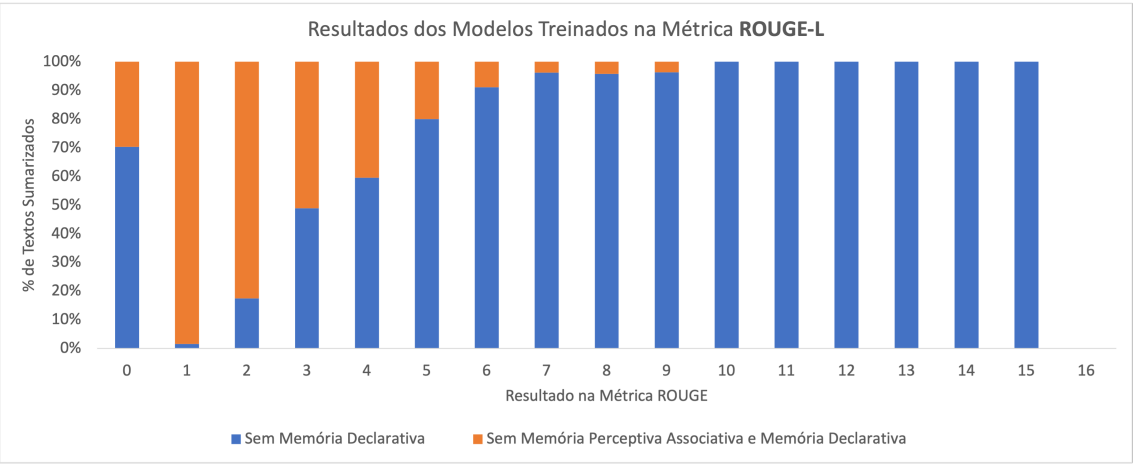
Fonte: autora.

Figura 48 – Gráfico com a distribuição das pontuações obtidas na ROUGE-2 pelos modelos treinados: sem Memória Declarativa (em azul) e sem Memória Perceptiva Associativa e Memória Declarativa (em laranja).



Fonte: autora.

Figura 49 – Gráfico com a distribuição das pontuações obtidas na ROUGE-L pelos modelos treinados: sem Memória Declarativa (em azul) e sem Memória Perceptiva Associativa e Memória Declarativa (em laranja).



Fonte: autora.

6 CONCLUSÃO

O presente trabalho propôs desenvolver um modelo de sumarização abstrativa de texto utilizando como base a estrutura teórica do LIDA (Seção 3.4). Para isso, foram utilizados conceitos encontrados na literatura aplicados na construção da sumarização abstrativa (Capítulo 3). Esses conceitos foram interpretados juntos aos conceitos teóricos dos módulos do LIDA e então utilizados na construção do modelo proposto (Capítulo 4).

Foram realizados experimentos para ajudar na análise de desempenho do modelo proposto. O primeiro experimento foi realizado para avaliar o desempenho do *Word2Vec*, alocado na Memória Perceptiva Associativa e da redução de dimensão do *Word2Vec* aplicada na Memória Declarativa, construído com diferentes dimensões. Esse experimento capturou os resultados referentes à semântica e sintaxe das palavras. Os demais experimentos avaliaram a assertividade do modelo proposto na geração da sumarização abstrativa de texto.

Assim, foram realizados quatro experimentos em que um módulo era descartado e então treinado por vez; ou seja, quatro modelos diferentes foram treinados para compreender a influência dos módulos da arquitetura do LIDA no resultado da sumarização.

Com os resultados obtidos dos experimentos alguns pontos importantes foram levantados.

Nos experimentos realizados na Seção 5.1.2, que verifica a qualidade do embedding gerado pela redução de dimensão e Seção 5.2.2, que retira a Memória Declarativa a qual utiliza a técnica de redução, foram obtidos resultados ruins dos embeddings após o uso da técnica desenvolvida por (RAUNAK; GUPTA; METZE, 2019) e detalhada na Seção 3.3.2, o que comprova que a técnica não é efetiva e portanto, é necessário que mais estudos sejam realizados para melhor desenvolvê-la.

Na Seção 5.2.3 foram apresentados os resultados obtidos do experimento que retirou a Memória Perceptiva Associativa a qual utiliza o *Word2Vec*. Quando a memória foi retirada e treinada junto ao codificador (Codeletes Construtores de Estruturas) e decodificador (Memória Processual), o desempenho do modelo foi muito menor, quando comparado ao modelo implementado com a Memória Perceptiva Associativa na qual o *embedding* foi treinado a parte. Isso indica que existe um ganho no treinamento de forma apartada do *embedding*.

Na Seção 5.2.4 os Módulos Codeletes de Atenção e Espaço de Trabalho Global foram retirados e os resultados demonstraram que, esses dois módulos são essenciais para o modelo

gerar as sentenças baseada no texto de entrada, sem essa técnica, o modelo gera de maneira aleatória as palavras do resumo.

No experimento realizado na Seção 5.2.5 foi identificada uma fragilidade na métrica ROUGE, o modelo gerou a mesma sequência contendo pronomes e artigos de forma repetida e a métrica gerou resultados bons, transmitindo uma falsa impressão de que o resumo gerado estava correto. Sendo assim, é importante que haja uma checagem na coesão e coerência do texto gerado, ainda que não seja automática a verificação, para garantir a possibilidade de leitura do texto gerado e não apenas atingir altas pontuações na métrica.

Por fim, foi observado através do experimento demonstrado na Seção 5.2.6 que o modelo apresentado na Seção 5.2.2 obteve o melhor resultado. Esse modelo foi executado com os módulos: Memória Sensorial, Memória Perceptiva Associativa, Codeletes Construtores de Estruturas, Codeletes de Atenção, Espaço de Trabalho Global e Memória Processual e, portanto, essa é a arquitetura final proposta neste trabalho.

Conforme apresentado no Capítulo 5, os modelos treinados não atingiram resultados próximos aos modelos estudados no Capítulo 2. Há uma complexidade no treinamento da arquitetura, principalmente no que se refere a capacidade computacional necessária para obter bons resultados, assim como o ajuste dos hiperparâmetros; ou seja, para modelos com a complexidade semelhante ao proposto neste trabalho, a configuração e definição do número de camadas, valor de regularização, taxa de aprendizado, número de tokens escolhidos e etc, são parâmetros decisivos para o bom desempenho, porém, a necessidade de adaptar à capacidade das máquinas utilizadas, torna a tarefa ainda mais difícil. Sendo assim, uma forma de melhorar os resultados do modelo proposto pode ser utilizar redes pré-treinadas e executar o *fine-tuning* das mesmas, uma vez que essas redes já possuem uma arquitetura definida e já foram executadas com sucesso.

Finalmente, o presente trabalho traz como contribuição uma nova perspectiva de construção de sumarização de texto. O LIDA é um modelo cognitivo baseado em teorias desenvolvidas pela psicologia e neurociência, que estudam o funcionamento da cognição humana, isto é, como funciona o processo de aprendizado humano. Essas teorias são transpostas em regras lógicas computacionais no LIDA, e, por esse motivo, auxilia no desenho da arquitetura de tarefas e processos que são facilmente desenvolvidos por humanos, mas que apresentam uma complexidade na adaptação em modelos computacionais. Uma dessas tarefas é justamente a sumarização de texto, por isso, o LIDA pode ser utilizado como base para adaptá-las. Cada módulo tem uma função que, ao ser interpretada, transforma-se num passo que representa um

algoritmo ou técnica, o qual irá compor uma solução maior de uma tarefa, como foi mostrado na adaptação deste trabalho.

Para trabalhos futuros, entende-se que o uso de transformers possa ajudar, principalmente na interação dos módulos Codeletes Construtores de Estruturas e Memória Processual, devido à diferente abordagem empregada no mecanismo de atenção. Além disso, há uma necessidade na literatura de trabalhar com a abordagem de sumarização de texto aplicada para língua portuguesa, o que poderia ser uma contribuição o desenvolvimento da arquitetura para uma língua diferente da inglesa.

REFERÊNCIAS

AGIRRE, Eneko et al. A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches. In: PROCEEDINGS of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Boulder, Colorado: Association for Computational Linguistics, jun. 2009. P. 19–27. Disponível em: <https://aclanthology.org/N09-1003>.

AGRAWAL, Pulin; FRANKLIN, Stan; SNAIDER, Javier. Sensory Memory for Grounded Representations in a Cognitive Architecture. In:

ALVAREZ, Jon Ezeiza; BAST, Hannah. A review of word embedding and document similarity algorithms applied to academic text. **bachelor thesis**, university of freiburg, 2017.

ANDO, Rie Kubota; ZHANG, Tong. A framework for learning predictive structures from multiple tasks and unlabeled data. **Journal of Machine Learning Research**, v. 6, Nov, p. 1817–1853, 2005.

BAARS, Bernard J. A cognitive theory of consciousness. In:

_____. In the Theater of Consciousness: The Workspace of the Mind. In:

BAHDANAU, Dzmitry; CHO, Kyunghyun; BENGIO, Yoshua. Neural machine translation by jointly learning to align and translate. **arXiv preprint arXiv:1409.0473**, 2014.

BANKO, Michele; MITTAL, Vibhu O.; WITBROCK, Michael J. Headline Generation Based on Statistical Translation. In: ACL. 2000.

BARRIOS, Federico et al. Variations of the similarity function of textrank for automated summarization. **arXiv preprint arXiv:1602.03606**, 2016.

BENGIO, Yoshua et al. A neural probabilistic language model. **Journal of machine learning research**, v. 3, Feb, p. 1137–1155, 2003.

BLITZER, John; MCDONALD, Ryan; PEREIRA, Fernando. Domain adaptation with structural correspondence learning. In: PROCEEDINGS of the 2006 conference on empirical methods in natural language processing. 2006. P. 120–128.

BOJANOWSKI, Piotr et al. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

BROWN, Peter F et al. Class-based n-gram models of natural language. **Computational linguistics**, v. 18, n. 4, p. 467–480, 1992.

CAO, Ziqiang et al. Retrieve, rerank and rewrite: Soft template based neural summarization. In: PROCEEDINGS of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2018. P. 152–161.

CELIKYILMAZ, Asli et al. Deep communicating agents for abstractive summarization. **arXiv preprint arXiv:1803.10357**, 2018.

CHEN, Yen-Chun; BANSAL, Mohit. Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting. **ArXiv**, abs/1805.11080, 2018.

CHENG, Jianpeng; LAPATA, Mirella. Neural summarization by extracting sentences and words. **arXiv preprint arXiv:1603.07252**, 2016.

CHOLLET, Francois et al. **Keras**. 2015. Disponível em: <https://github.com/fchollet/keras>.

CHOPRA, Sumit; AULI, Michael; RUSH, Alexander M. Abstractive sentence summarization with attentive recurrent neural networks. In: PROCEEDINGS of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016. P. 93–98.

CLARKE, James; LAPATA, Mirella. Global inference for sentence compression: An integer linear programming approach. **Journal of Artificial Intelligence Research**, v. 31, p. 399–429, 2008.

COHAN, Arman; GOHARIAN, Nazli. Scientific article summarization using citation-context and article's discourse structure. **arXiv preprint arXiv:1704.06619**, 2017.

COHN, Trevor; LAPATA, Mirella. Sentence Compression Beyond Word Deletion. In: COLING. 2008.

COLLOBERT, Ronan; WESTON, Jason. A unified architecture for natural language processing: Deep neural networks with multitask learning. In: PROCEEDINGS of the 25th international conference on Machine learning. 2008. P. 160–167.

DEERWESTER, Scott et al. Indexing by latent semantic analysis. **Journal of the American society for information science**, Wiley Online Library, v. 41, n. 6, p. 391–407, 1990.

DEVLIN, Jacob et al. Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

EDMUNDSON, H. P. New Methods in Automatic Extracting. **J. ACM**, Association for Computing Machinery, New York, NY, USA, v. 16, n. 2, p. 264–285, abr. 1969. Disponível em: <https://doi.org/10.1145/321510.321519>.

FRANKLIN, Stan. A conscious artifact? **Journal of Consciousness Studies**, Imprint Academic, v. 10, n. 4-5, p. 47–66, 2003.

_____. Modeling consciousness and cognition in software agents. In: CITESEER. PROCEEDINGS of the Third International Conference on Cognitive Modeling, Groeningen, NL. 2000.

FRANKLIN, Stan; KELEMEN, Arpad; MCCAULEY, Lee. IDA: A cognitive agent architecture. In: IEEE. SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218). 1998. v. 3, p. 2646–2651.

FRANKLIN, Stan; PATTERSON JR, FG. The LIDA architecture: Adding new modes of learning to an intelligent, autonomous, software agent. **pat**, v. 703, p. 764–1004, 2006.

FRANKLIN, Stan et al. A LIDA cognitive model tutorial. **Biologically Inspired Cognitive Architectures**, Elsevier, v. 16, p. 105–130, 2016.

FRANKLIN, Stan et al. Lida: A computational model of global workspace theory and developmental learning., 2007.

FRANKLIN, Stan et al. LIDA: A Systems-level Architecture for Cognition, Emotion, and Learning. **IEEE Transactions on Autonomous Mental Development**, v. 6, p. 19–41, 2014.

FRANKLIN, Stan et al. LIDA: A systems-level architecture for cognition, emotion, and learning. **IEEE Transactions on Autonomous Mental Development**, IEEE, v. 6, n. 1, p. 19–41, 2013.

G1. **Segunda-feira, 30 de novembro**. 2020. https://g1.globo.com/resumo-do-dia/noticia/2020/11/30/segunda-feira-30-de-novembro.ghtml, último acesso em 01/12/2020.

GEHRMANN, Sebastian; DENG, Yuntian; RUSH, Alexander M. Bottom-up abstractive summarization. **arXiv preprint arXiv:1808.10792**, 2018.

GONG, Hongyu et al. Preposition Sense Disambiguation and Representation. In: PROCEEDINGS of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018. P. 1510–1521.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. MIT press, 2016.

GUO, Han; PASUNURU, Ramakanth; BANSAL, Mohit. Soft Layer-Specific Multi-Task Summarization with Entailment and Question Generation. In: ACL. 2018.

HAHN, Michael; KELLER, Frank. Modeling human reading with neural attention. **arXiv preprint arXiv:1608.05604**, 2016.

HERMANN, Karl Moritz et al. Teaching machines to read and comprehend. In: ADVANCES in neural information processing systems. 2015. P. 1693–1701.

HILL, Felix; REICHART, Roi; KORHONEN, Anna. SimLex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. **CoRR**, abs/1408.3456, 2014. arXiv: 1408.3456. Disponível em: <http://arxiv.org/abs/1408.3456>.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HU, Baotian; CHEN, Qingcai; ZHU, Fangze. Lcsts: A large scale chinese short text summarization dataset. **arXiv preprint arXiv:1506.05865**, 2015.

JING, Hongyan. Sentence reduction for automatic text summarization. In: SIXTH Applied Natural Language Processing Conference. 2000. P. 310–315.

JOHNSON, Frances C et al. The application of linguistic processing to automatic abstract generation. Morgan Kauffman, 1997.

JOHNSON, John L; CAULFIELD, H John; TAYLOR, Jaime R. Artificial consciousness algorithm for an autonomous system. In: IEEE. PROCEEDINGS of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium. 2000. v. 5, p. 635–640.

KEDZIE, Chris; MCKEOWN, Kathleen; DAUME III, Hal. Content selection in deep learning models of summarization. **arXiv preprint arXiv:1810.12343**, 2018.

KHAYI, Nisrine Ait; FRANKLIN, Stan. Initiating language in LIDA: Learning the meaning of vervet alarm calls. **Biologically Inspired Cognitive Architectures**, Elsevier, v. 23, p. 7–18, 2018.

KIM, Byeongchang; KIM, Hyunwoo; KIM, Gunhee. Abstractive summarization of reddit posts with multi-level memory networks. **arXiv preprint arXiv:1811.00783**, 2018.

KIM, Eric. **Demystifying Neural Network in Skip-Gram Language Modeling**. 2019. https://aegis4048.github.io/demystifying_neural_network_in_skip_gram_language_modeling, último acesso em 13/09/2020.

KINGMA, Diederik P; WELLING, Max. Auto-encoding variational bayes. **arXiv preprint arXiv:1312.6114**, 2013.

KNIGHT, Kevin; MARCU, Daniel. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. **Artificial Intelligence**, Elsevier, v. 139, n. 1, p. 91–107, 2002.

KRYŚCIŃSKI, Wojciech et al. Improving abstraction in text summarization. **arXiv preprint arXiv:1808.07913**, 2018.

KUGELE, Sean; FRANKLIN, Stan. A Study in Activation: Towards a Common Lexicon and Functional Taxonomy in Cognitive Architectures, 2020.

_____. Learning In LIDA, 2020.

KUPIEC, Julian; PEDERSEN, Jan; CHEN, Francine. A trainable document summarizer. In: PROCEEDINGS of the 18th annual international ACM SIGIR conference on Research and development in information retrieval. 1995. P. 68–73.

LAPLACE, Pierre Simon. **Théorie analytique des probabilités**. Courcier, 1820.

LI, Chenliang et al. Guiding generation for abstractive text summarization based on key information guide network. In: PROCEEDINGS of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers). 2018. P. 55–60.

LI, Piji; BING, Lidong; LAM, Wai. Actor-critic based training framework for abstractive summarization. **arXiv preprint arXiv:1803.11070**, 2018.

LI, Piji et al. Deep recurrent generative decoder for abstractive text summarization. **arXiv preprint arXiv:1708.00625**, 2017.

LIN, Chin-Yew. Rouge: A package for automatic evaluation of summaries. In: TEXT summarization branches out. 2004. P. 74–81.

LIN, Junyang et al. Global encoding for abstractive summarization. **arXiv preprint arXiv:1805.03989**, 2018.

LIU, He; YU, Hongliang; DENG, Zhi-Hong. Multi-document summarization based on two-level sparse representation model. In: TWENTY-NINTH AAAI conference on artificial intelligence. 2015.

LUHN, Hans Peter. The automatic creation of literature abstracts. **IBM Journal of research and development**, Ibm, v. 2, n. 2, p. 159–165, 1958.

LUND, Kevin; BURGESS, Curt. Producing high-dimensional semantic spaces from lexical co-occurrence. **Behavior research methods, instruments, & computers**, Springer, v. 28, n. 2, p. 203–208, 1996.

LUONG, Minh-Thang; PHAM, Hieu; MANNING, Christopher D. Effective approaches to attention-based neural machine translation. **arXiv preprint arXiv:1508.04025**, 2015.

MA, Shuming; SUN, Xu. A semantic relevance based neural network for text summarization and text simplification. **arXiv preprint arXiv:1710.02318**, 2017.

MA, Shuming et al. Improving semantic relevance for sequence-to-sequence learning of chinese social media text summarization. **arXiv preprint arXiv:1706.02459**, 2017.

MACHADO, Lívia et al. **Governo de SP anuncia recuo e coloca todo o estado na fase amarela do plano de flexibilização**. 2020. https://g1.globo.com/sp/sao-paulo/noticia/2020/11/30/governo-de-sp-anuncia-recuo-e-coloca-todo-o-estado-na-fase-amarela-do-plano-de-flexibilizacao.ghtml, último acesso em 01/12/2020.

MARTÍN ABADI et al. **TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems**. 2015. Software available from tensorflow.org. Disponível em: <https://www.tensorflow.org/>.

MCCALL, Ryan James et al. Artificial Motivation for Cognitive Software Agents. **Journal of Artificial General Intelligence**, v. 11, p. 38–69, 2020.

MCDONALD, Ryan. Discriminative sentence compression with soft syntactic evidence. In: 11TH Conference of the European Chapter of the Association for Computational Linguistics. 2006.

MIKOLOV, Tomas et al. Distributed representations of words and phrases and their compositionality. In: ADVANCES in neural information processing systems. 2013. P. 3111–3119.

MIKOLOV, Tomas et al. **Efficient Estimation of Word Representations in Vector Space**. arXiv, 2013. Disponível em: <https://arxiv.org/abs/1301.3781>.

_____. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MORADI, Milad; GHADIRI, Nasser. Different approaches for identifying important concepts in probabilistic biomedical text summarization. **Artificial intelligence in medicine**, Elsevier, v. 84, p. 101–116, 2018.

MORENO, Raúl Arrabales; ESPINO, Agapito Ledezma; DE MIGUEL, Araceli Sanchis. Modeling consciousness for autonomous robot exploration. In: SPRINGER. INTERNATIONAL Work-Conference on the Interplay Between Natural and Artificial Computation. 2007. P. 51–60.

NALLAPATI, Ramesh et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. **arXiv preprint arXiv:1602.06023**, 2016.

NALLAPATI, Ramesh; ZHAI, Feifei; ZHOU, Bowen. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In: THIRTY-FIRST AAAI Conference on Artificial Intelligence. 2017.

NAPOLES, Courtney; GORMLEY, Matthew R; VAN DURME, Benjamin. Annotated gigaword. In: PROCEEDINGS of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX). 2012. P. 95–100.

NARAYAN, Shashi et al. Neural extractive summarization with side information. **arXiv preprint arXiv:1704.04530**, 2017.

NELSON, Stuart J et al. Unified Medical Language System®(UMLS®) Project. In: ENCYCLOPEDIA of library and information sciences. 2010. P. 5320–5327.

NEMA, Preksha et al. Diversity driven attention model for query-based abstractive summarization. **arXiv preprint arXiv:1704.08300**, 2017.

OLAH, Christopher. **Understanding LSTM Networks**. 2015. http://colah.github.io/posts/2015-08-Understanding-LSTMs, último acesso em 06/09/2020.

PAICE, Chris D. Constructing literature abstracts by computer: Techniques and prospects. **Inf. Process. Manag.**, v. 26, n. 1, p. 171–186, 1990.

PAULUS, Romain; XIONG, Caiming; SOCHER, Richard. A deep reinforced model for abstractive summarization. **arXiv preprint arXiv:1705.04304**, 2017.

PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher D. Glove: Global vectors for word representation. In: PROCEEDINGS of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014. P. 1532–1543.

PRASAD, Dilip K; STARZYK, Janusz A. A perspective on machine consciousness. In: CITESEER. INTL. Conf. Advanced Cognitive Technologies and Applications. 2010. P. 109–114.

RAUNAK, Vikas; GUPTA, Vivek; METZE, Florian. Effective dimensionality reduction for word embeddings. In: PROCEEDINGS of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019). 2019. P. 235–243.

REZENDE, Danilo Jimenez; MOHAMED, Shakir; WIERSTRA, Daan. Stochastic backpropagation and approximate inference in deep generative models. **arXiv preprint arXiv:1401.4082**, 2014.

RONG, Xin. word2vec parameter learning explained. **arXiv preprint arXiv:1411.2738**, 2014.

ROSSIELLO, Gaetano; BASILE, Pierpaolo; SEMERARO, Giovanni. Centroid-based text summarization through compositionality of word embeddings. In: PROCEEDINGS of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres. 2017. P. 12–21.

RUDRA, Koustav et al. Summarizing situational tweets in crisis scenario. In: PROCEEDINGS of the 27th ACM Conference on Hypertext and Social Media. 2016. P. 137–147.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, v. 323, p. 533–536, 1986.

RUSH, Alexander M; CHOPRA, Sumit; WESTON, Jason. A neural attention model for abstractive sentence summarization. **arXiv preprint arXiv:1509.00685**, 2015.

SAGGION, Horacio; POIBEAU, Thierry. Automatic text summarization: Past, present and future. In: MULTI-SOURCE, multilingual information extraction and summarization. Springer, 2013. P. 3–21.

SAK, Hasim; SENIOR, Andrew W; BEAUFAYS, Françoise. Long short-term memory recurrent neural network architectures for large scale acoustic modeling, 2014.

SANDHAUS, Evan. The new york times annotated corpus. **Linguistic Data Consortium, Philadelphia**, v. 6, n. 12, e26752, 2008.

SCHUTZE, Hinrich. Dimensions of meaning. In: IEEE. SUPERCOMPUTING'92: Proceedings of the 1992 ACM/IEEE Conference on Supercomputing. 1992. P. 787–796.

SEE, Abigail; LIU, Peter J; MANNING, Christopher D. Get to the point: Summarization with pointer-generator networks. **arXiv preprint arXiv:1704.04368**, 2017.

SHI, Tian et al. Neural abstractive text summarization with sequence-to-sequence models. **arXiv preprint arXiv:1812.02303**, 2018.

SONG, Kaiqiang; ZHAO, Lin; LIU, Fei. Structure-infused copy mechanisms for abstractive summarization. **arXiv preprint arXiv:1806.05658**, 2018.

SONG, Shengli; HUANG, Haitao; RUAN, Tongxiao. Abstractive text summarization using LSTM-CNN based deep learning. **Multimedia Tools and Applications**, Springer, v. 78, n. 1, p. 857–875, 2019.

STANDARDS, National Institute of; (NIST), Technology. **DUC 2002 GUIDELINES**. 2002. https://www-nlpir.nist.gov/projects/duc/guidelines/2002.html, último acesso em 16/07/2020.

STARZYK, Janusz A.; PRASAD, Dilip K. A computational model of machine consciousness. **International Journal of Machine Consciousness**, v. 03, p. 255–281, 2011.

SUN, Ron. Accounting for the Computational Basis of Consciousness: A Connectionist Approach. **Consciousness and Cognition**, v. 8, p. 529–565, 1999.

_____. Learning, action and consciousness: A hybrid approach toward modelling consciousness. **Neural Networks**, Elsevier, v. 10, n. 7, p. 1317–1331, 1997.

SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. In: ADVANCES in neural information processing systems. 2014. P. 3104–3112.

TAI, Kai Sheng; SOCHER, Richard; MANNING, Christopher D. Improved semantic representations from tree-structured long short-term memory networks. **arXiv preprint arXiv:1503.00075**, 2015.

TAN, Jiwei; WAN, Xiaojun; XIAO, Jianguo. Abstractive document summarization with a graph-based attentional neural model. In: PROCEEDINGS of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2017. P. 1171–1181.

TAYLOR, John G. CODAM: A neural network model of consciousness. **Neural Networks**, Elsevier, v. 20, n. 9, p. 983–992, 2007.

TONONI, Giulio. The integrated information theory of consciousness: an updated account. **Archives italiennes de biologie**, v. 150, n. 2/3, p. 56–90, 2012.

VASSEV, Emil; HINCHEY, Mike. Implementing artificial awareness with knowlang. In: IEEE. 2013 IEEE International Systems Conference (SysCon). 2013. P. 580–586.

WANG, Li et al. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. **arXiv preprint arXiv:1805.03616**, 2018.

WOODSEND, Kristian; FENG, Yansong; LAPATA, Mirella. Title Generation with Quasi-Synchronous Grammar. In: EMNLP. 2010.

YASUNAGA, Michihiro et al. Graph-based neural multi-document summarization. **arXiv preprint arXiv:1706.06681**, 2017.

YOGATAMA, Dani; LIU, Fei; SMITH, Noah A. Extractive summarization by maximizing semantic volume. In: PROCEEDINGS of the 2015 Conference on Empirical Methods in Natural Language Processing. 2015. P. 1961–1966.

YOUSEFI-AZAR, Mahmood; HAMEY, Len. Text summarization using unsupervised deep learning. **Expert Systems with Applications**, Elsevier, v. 68, p. 93–105, 2017.

ZAJIC, David M.; DORR, Bonnie J.; SCHWARTZ, Richard M. BBN/UMD at DUC-2004: Topiary. In:

ZENG, Wenyuan et al. Efficient summarization with read-again and copy mechanism. **arXiv preprint arXiv:1611.03382**, 2016.

ZHOU, Qingyu et al. Neural document summarization by jointly learning to score and select sentences. **arXiv preprint arXiv:1807.02305**, 2018.