**UNIVERSIDADE ESTADUAL DO CEARÁ**

**CENTRO DE CIÊNCIAS E TECNOLOGIA**

**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

**MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO**

**ANTONIO ALEX DE SOUZA**

# LUPPAR NEWS-REC: UM RECOMENDADOR INTELIGENTE DE NOTÍCIAS

**FORTALEZA – CEARÁ**

**2019**

ANTONIO ALEX DE SOUZA

# LUPPAR NEWS-REC: UM RECOMENDADOR INTELIGENTE DE NOTÍCIAS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Orientador: Prof. Dr. José Everardo Bessa Maia

FORTALEZA – CEARÁ

2019

Dados Internacionais de Catalogação na Publicação

Universidade Estadual do Ceará

Sistema de Bibliotecas

Souza, Antonio Alex de.
Luppar news-rec: um recomendador inteligente de notícias [recurso eletrônico] / Antonio Alex de Souza. - 2019.
1 CD-ROM: il.; 4 ¾ pol.

CD-ROM contendo o arquivo no formato PDF do trabalho acadêmico com 94 folhas, acondicionado em caixa de DVD Slim (19 x 14 cm x 7 mm).

Dissertação (mestrado acadêmico) - Universidade Estadual do Ceará, Centro de Ciências e Tecnologia, Mestrado Acadêmico em Ciência da Computação, Fortaleza, 2019.
Área de concentração: Ciência da Computação.
Orientação: Prof. Dr. José Everardo Bessa Maia.

1. Classificação. 2. Representação de Documentos. 3. Incorporação de Palavras. 4. Notícias Curtas. 5. Recomendação. I. Título.

ANTONIO ALEX DE SOUZA

LUPPAR NEWS-REC: UM RECOMENDADOR INTELIGENTE DE NOTÍCIAS

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação

Aprovada em: 24 de Setembro de 2019

BANCA EXAMINADORA

Prof. Dr. José Everardo Bessa Maia (Orientador)
Universidade Estadual do Ceará – UECE

Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará - UFC

Prof. PhD. Matheus Henrique Esteves Paixão
Universidade Estadual do Ceará - UECE

À minha Esposa, por todo incentivo, apoio e compreensão durante essa caminhada, sempre mostrando que nada é impossível, basta querer.
À minha Família, por sempre acreditar em mim e me apoiar em todas minhas decisões.

## AGRADECIMENTOS

A Deus que permitiu que mais esse sonho fosse concretizado e por tudo que já conquistei com muito esforço ao longo da minha vida.

Ao meu orientador Prof. Dr. José Everardo Bessa Maia pelas suas aulas, orientação, incentivo, paciência, atenção e dedicação oferecidas durante a construção desse trabalho.

A minha esposa Lídia Araújo Santos por seu incentivo, apoio e amor, pois foi o que me deu a força necessária para seguir em frente. Além da compreensão da minha ausência em virtude do tempo empregado aos estudos, concedendo assim a oportunidade da realização de mais esse sonho. Se não fosse por ela teria postergado ainda mais e talvez não teria entrado no mestrado.

Aos meus pais pelo apoio e incentivo de sempre em todas as áreas da minha vida. Obrigado pela excelente educação que me deram. Aos meus irmãos pela ajuda constante.

A minha avó Zélia, uma pessoa incrível e sábia, que apesar da distância física, está sempre aqui comigo. E ao meu avô Raimundo (*in memoriam*), que tanta falta nos faz. Exemplo de mansidão, carinho e amor ao próximo. Sei que de onde estiver, estará muito feliz e realizado com mais essa conquista do seu neto. Aos meus avós Francisca e Sebastião (*in memoriam*), que apesar de eu não ter tido convivência, nos poucos momentos juntos me ensinaram muito com exemplos.

A minha Filha, que apesar de não termos convivência, que compreenda minha ausência em muitos momentos importantes.

Aos meus familiares: tios(as), primos(as), cunhados(as), sogro, por sempre me apoiarem e entenderem meu isolamento em inúmeros finais de semanas.

Aos meus professores por me proporcionarem o conhecimento não apenas racional, mas a manifestação do caráter e afetividade da educação no processo de formação profissional.

Aos meus amigos das seguintes áreas que me apoiaram e incentivaram dia-a-dia: TI, primeiramente pelo apoio total do André, Renan, Régis e do Joel por ser "o cobaia"dos testes de apresentação, além de todos os outros que de alguma forma me ajudaram a alcançar esse objetivo; Mestrado, em especial, que me ajudaram muito principalmente em programação, pois minha área de atuação era mais dados, fora o tempo longe do ambiente totalmente acadêmico. A todos meu muito obrigado por permitirem que esta dissertação seja uma realidade.

"É melhor lançar-se à luta em busca do triunfo mesmo expondo-se ao insucesso, que formar fila com os pobres de espírito, que nem gozam muito nem sofrem muito; E vivem nessa penumbra cinzenta sem conhecer nem vitoria nem derrota."

(Franklin Roosevelt)

## RESUMO

A presente dissertação documenta a construção de um sistema recomendador de notícias, denominado ***Luppar News-Rec***, e sua avaliação. ***Luppar News-Rec*** é constituído de três subsistemas: o subsistema de captura, pré-processamento e armazenamento, o subsistema de classificação e o subsistema de aquisição de perfis de assinantes e distribuição de notícias. O foco principal é no subsistema de classificação de notícias sendo que para os outros dois foram implementadas apenas funcionalidades básicas essenciais. No projeto do subsistema de classificação avalia-se a eficácia da combinação de modelos de representação de documentos, incluindo uma abordagem *embedding* ponderada, com algoritmos de classificação de texto monorrótulo e multirrótulo. O trabalho deste subsistema é classificar as notícias em tópicos de interesse e distribuí-las conforme o perfil dos assinantes. Os modelos de representação de documentos do tipo BoW, matriz TF-IDF, *Word2Vec* e *FastText*, esses dois últimos também utilizando a abordagem E2V-IDF, são combinados com os classificadores KNN, SVM(RBF), *Decision Tree*, *Random Forest*. Os resultados mostram o desempenho significativamente superior da combinação SVM (kernel: RBF) com *embeddings* utilizando a abordagem `E2V-IDF` em relação a representações *embeddings* tradicionais, para a maioria das coleções abordadas.



**Palavras-chave:** Classificação. Representação de Documentos. Incorporação de Palavras. Notícias. Recomendação

## ABSTRACT

The present dissertation documents the construction of a news recommending system, called Luppar News-Rec, and its evaluation. Luppar News-Rec consists of three subsystems: the capture, preprocessing and storage subsystem, the rating subsystem, and the subscriber profiling and news distribution subsystem. The main focus is on the news rating subsystem, and for the others only essential functionalities have been implemented. The design of the classification subsystem assesses the effectiveness of combining document representation models, including a weighted embedding approach, with single-label and multi-label text classification algorithms. The role of this subsystem is to classify the news into topics of interest and distribute them according to the subscriber profile. BoW, TF-IDF matrix, Word2Vec, and FastText document representation models, the latter two also using the E2V-IDF approach, are combined with the KNN, SVM (RBF), Decision Tree, Random Forest classifiers. The results show the significantly superior performance of combining SVM (kernel: RBF) with embeddings using the E2V-IDF approach over traditional embeddings representations for most of the collections addressed.



**Keywords:** Classification. Document Representation. Word Embeddings. News. Recommender

## LISTA DE ILUSTRAÇÕES

**Figura 1 – Diagrama em blocos de um Sistema Recomendador de Notícias (SRN)** . 16
**Figura 2 – Visualização tabular do *Bag-of-Words (BoW)*** . . . . . 22
**Figura 3 – Visualização tabular do *Term Frequency-Inverse Document Frequency* (TF-IDF)** . . . . . 24
**Figura 4 – *Embedding* - *Word2Vec* (Palavra: Toyota) - Utilizando *t-Distributed Stochastic Neighbor Embedding* (t-SNE), uma técnica para redução de dimensionalidade que é particularmente adequada para a visualização de conjuntos de dados de alta dimensão** . . . . . 25
**Figura 5 – Arquitetura do modelo *Skip-gram*** . . . . . 26
**Figura 6 – Arquitetura do modelo *CBOW*** . . . . . 28
**Figura 7 – Visualização tabular de uma representação *Embedding* de documentos** 29
**Figura 8 – Pseudocódigo da Representação de Documento** `E2V-IDF` . . . . . 30
**Figura 9 – Classificação de uma nova amostra através do método KNN, a figura apresenta dois valores para K (K=3 e K=5), para o K=3, é calculado a menor distância entre a nova amostra e seus três vizinhos mais próximos, podemos visualizar pela figura que a nova amostra será atribuída à Classe 2** . . . . . 35
**Figura 10 – Cálculo da distância $d$ entre os hiperplanos H1 e H2** . . . . . 36
**Figura 11 – Análise do parâmetro $\gamma$ na função *kernel*: RBF** . . . . . 37
**Figura 12 – Visualização de um classificador utilizando árvore de decisão** . . . . . 39
**Figura 13 – Ilustração da lógica por trás do algoritmo *Random Forest*** . . . . . 41
**Figura 14 – Demonstração de comportamento - Curva ROC** . . . . . 45
**Figura 15 – Estrutura do SRN baseado em *Publish-Subscribe*, apresentando os três subsistemas que o compõe** . . . . . 55
**Figura 16 – Processo automático de coleta e transformação de notícias (EN / PT-BR)** 56
**Figura 17 – Interface para testes e demonstrações do Sistema Recomendador de Notícias *Luppar News Recommender (Luppar News-Rec)*** . . . . . 61
**Figura 18 – Área sob a Curva *Precision-Recall* com *Average Precision* de 87% para todos os tópicos da coleção Z5News (combinação: SVM(RBF)+FT (E2V-IDF))** . . . . . 71
**Figura 19 – Curva ROC média da coleção Z5News (*SVM(RBF)+FT* (`E2V-IDF`))** . . 71

**Figura 20 – Curva ROC detalhada por tópico para a coleção Z5News (combinação: *SVM(RBF)+FT* (E2V-IDF))** . . . . . 72
**Figura 21 – Área sob a Curva *Precision-Recall* com *Average Precision* de 87% para todos os tópicos da coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))** . . . . . 75
**Figura 22 – Curva ROC média da coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))** . . . . . 76
**Figura 23 – Curva ROC detalhada por tópico para a coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))** . . . . . 76
**Figura 24 – Área sob a Curva *Precision-Recall* com *Average Precision* de 74% para todos os tópicos da coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))** . . . . . 78
**Figura 25 – Curva ROC média da coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))** . . . . . 79
**Figura 26 – Curva ROC detalhada por tópico para a coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))** . . . . . 79

## LISTA DE TABELAS


**Tabela 1 – Matriz de Confusão para classificação binária** . . . . . 42
**Tabela 2 – Características das coleções de notícias próprias** . . . . . 58
**Tabela 3 – Características das coleções de notícias de referência** . . . . . 63
**Tabela 4 – Bibliotecas da linguagem Python utilizadas no projeto** . . . . . 64
**Tabela 5 – Parametrização dos algoritmos utilizados durante os experimentos** . . . 65
**Tabela 6 – Características dos vetores *embeddings* pré-treinados (Inglês e Português)** 65
**Tabela 7 – Resultados da métrica *F1-Score* - Média(Variância) para coleções de Referência utilizando as representações: BoW e TF-IDF** . . . . . 66
**Tabela 8 – Resultados da métrica *F1-Score* - Média(Variância) para coleções próprias utilizando as representações: BoW e TF-IDF** . . . . . 66
**Tabela 9 – Resultados da métrica *F1-Score* para coleções de referência utilizando as representações *embeddings*: *Word2Vec* e *FastText* - Treinadas na Coleção** . . . . . 67
**Tabela 10 – Resultados da métrica *F1-Score* para coleções próprias utilizando as representações: *Word2Vec* e *FastText*- Treinadas na Coleção** . . . . . 67
**Tabela 11 – Resultados da métrica *F1-Score* para coleções de referência utilizando as representações *embeddings*: *Word2Vec* e *FastText* - Pré-Treinadas (Wiki)** . . . . . 68
**Tabela 12 – Apresenta o *F1-Score* para coleções próprias utilizando representações *embedding*: *Word2Vec* e *FastText* - Pré-Treinado (Wiki + STIL)** . . . . . 68
**Tabela 13 – Comparativo das combinações mais bem avaliadas** . . . . . 70
**Tabela 14 – Comparativo de combinações para a Coleção Z5News (*Offline*)** . . . . . 74
**Tabela 15 – Comparativo de combinações para a Coleção Z5NewsBrasil (*Offline*)** . 74
**Tabela 16 – Comparativo de combinações para a Coleção Z12News (*Offline*)** . . . . 74
**Tabela 17 – Comparativo de combinações para a Coleção Z20News (*Offline*)** . . . . 74
**Tabela 18 – Comparativo de combinações para a Coleção Z5News (*Online*)** . . . . . 77
**Tabela 19 – Comparativo de combinações para a Coleção Z5NewsBrasil (*Online*)** . 77
**Tabela 20 – Comparativo de combinações para a Coleção Z12News (*Online*)** . . . . 77
**Tabela 21 – Comparativo de combinações para a Coleção Z20News (*Online*)** . . . . 78
**Tabela 22 – Matriz de Confusão por tópicos para a coleção Z5News (*Online*) utilizando a combinação: SVM(RBF)+FT (`E2V-IDF`)** . . . . . 80

**Tabela 23 – Métricas de Classificação baseada na matriz de confusão por tópicos para a coleção Z5News (*Online*) (combinação: SVM(RBF)+FT (E2V-IDF))** 80
**Tabela 24 – Tempo em minutos para geração dos Modelos obtidos pelas combinações mais bem avaliadas e aplicadas as coleções próprias para Testes *Online*** . . . . . 82
**Tabela A1 – Simulação de Perfis de Assinantes (SVM(RBF)+FT (E2V-IDF))** . . . . . 91
**Tabela B1 – Número de notícias para: Treinamento (Teste *Offline* | Teste *Online*)** . . 92
**Tabela C1 – Treinando e Testando com dados da mesma coleção (dados de teste não vistos no treinamento)** . . . . . 93
**Tabela C2 – Treinando com dados de uma coleção e Testando com dados de outra coleção (também foi incluso, nos testes, dados da mesma coleção de treinamento, dados esses não vistos no treinamento)** . . . . . 94

## LISTA DE SÍMBOLOS

| | |
|---|---|
| 20*News* | *20 Newsgroups* |
| *BoW* | *Bag-of-Word* |
| *DT* | *Decision Tree* |
| $E2V-IDF$ | *Embedding* ponderado por *IDF* |
| $FT$ | *FastText* |
| *IDF* | *Inverse Document Frequency* |
| *KNN* | *K-Nearest Neighbor* |
| *R*8 | *Reuters-21578 (R8)* |
| *R*52 | *Reuters-21578 (R52)* |
| *RF* | *Random Forest* |
| *RBF* | *Radial Basis Function* |
| *RCV*1 | *Reuters Corpus Volume 1* |
| *RI* | Recuperação de Informação |
| *SR* | Sistemas de Recomendação |
| *SRN* | Sistemas de Recomendação de Notícias |
| *SVM* | *Support Vector Machines* |
| $TF-IDF$ | *Term Frequency–Inverse Document Frequency* |
| *W*2*V* | *Word2Vec* |
| *Z*5*News* | Coleção própria (EN - 5 Tópicos) |
| *Z*5*NewsBrasil* | Coleção própria (PT-BR - 5 Tópicos) |
| *Z*12*News* | Coleção própria (EN - 12 Tópicos) |
| *Z*20*News* | Coleção própria (EN - 20 Tópicos) |

**SUMÁRIO**


**1 INTRODUÇÃO** . . . . . 16
1.1 OBJETIVOS . . . . . 18
1.2 METODOLOGIA . . . . . 18
1.3 ESTRUTURA DA DISSERTAÇÃO . . . . . 19
**2 FUNDAMENTAÇÃO TEÓRICA** . . . . . 20
2.1 REPRESENTAÇÕES DE DOCUMENTOS . . . . . 20
**2.1.1 Pré-processamento (Extração)** . . . . . 21
**2.1.2 Representações de Documentos** . . . . . 22
**2.1.3 A abordagem** `E2V-IDF` . . . . . 29
2.2 CLASSIFICAÇÃO DE TEXTOS . . . . . 31
**2.2.1 Uma caracterização da classificação de textos** . . . . . 32
**2.2.2 Algoritmos de classificação de textos** . . . . . 34
**2.2.3 Avaliação em Classificação** . . . . . 42
2.3 SISTEMAS DE RECOMENDAÇÃO . . . . . 46
**2.3.1 Sistema de Recomendação de Notícias** . . . . . 49
**3 TRABALHOS RELACIONADOS** . . . . . 51
**4 PROJETO DO SISTEMA** . . . . . 55
4.1 SUBSISTEMA DE CAPTURA, PRÉ-PROCESSAMENTO E ARMAZENAMENTO . . . . . 56
4.2 SUBSISTEMA DE CLASSIFICAÇÃO DE NOTÍCIAS (*CORE*) . . . . . 58
4.3 SUBSISTEMA DE AQUISIÇÃO DE PERFIS DE ASSINANTES E DISTRIBUIÇÃO . . . . . 59
**4.3.1 Interface** . . . . . 60
**5 AVALIAÇÃO, RESULTADOS E DISCUSSÃO** . . . . . 62
5.1 SELEÇÃO DA REPRESENTAÇÃO-CLASSIFICADOR . . . . . 62
**5.1.1 Testes preliminares** . . . . . 63
**5.1.2 Resultados da avaliação** . . . . . 64
**5.1.3 Discussão e Seleção** . . . . . 69
5.1.3.1 Seleção . . . . . 70
5.2 PLANO EXPERIMENTAL DE TESTES DO SISTEMA E MÉTRICAS . . 72
5.3 RESULTADOS . . . . . 73

**5.3.1** **Testes *Offline*** . . . . . 73
**5.3.2** **Testes *Online*** . . . . . 77
**5.3.3** **Teste *Online*: Avaliação por Tópico e por Assinante** . . . . . 80
5.4 DISCUSSÃO . . . . . 81
**6** **CONCLUSÃO** . . . . . 83
6.1 CONTRIBUIÇÕES DO TRABALHO . . . . . 83
6.2 LIMITAÇÕES . . . . . 84
6.3 TRABALHOS FUTUROS . . . . . 84
**REFERÊNCIAS** . . . . . 85
**APÊNDICES** . . . . . 90
APÊNDICE A – SIMULAÇÃO DE PERFIS DE ASSINANTES . . . . . 91
APÊNDICE B – COLEÇÕES PRÓPRIAS - DETALHAMENTO POR TÓPICOS . . . . . 92
APÊNDICE C – TESTE INTER-BASES . . . . . 93

# 1 INTRODUÇÃO

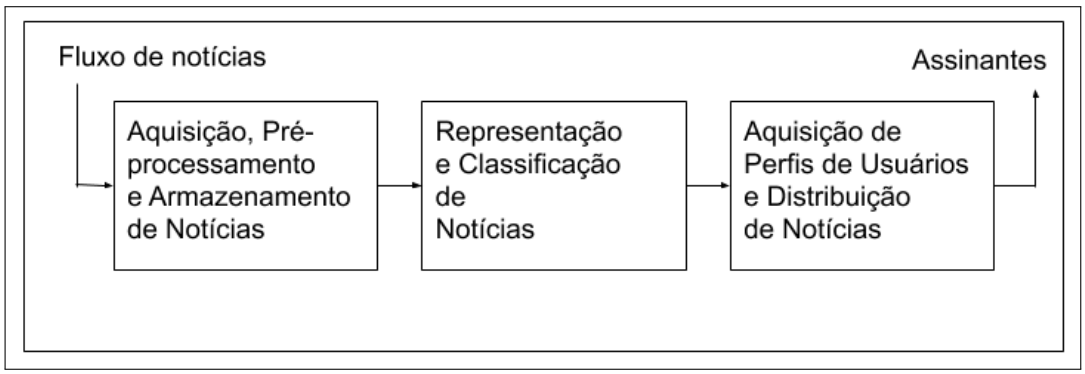
Com o grande e crescente volume de textos e notícias disponíveis na internet, torna-se difícil para um leitor encontrar as notícias de seu interesse sem o auxílio de ferramentas computacionalmente apropriadas. A criação destas ferramentas e seus algoritmos são assim um dos desafios atuais para reduzir o tempo e melhorar a precisão no acesso à informação.

Para auxiliar na busca mais eficiente por informações, um usuário (leitor) pode utilizar um Sistema de Recomendação (SR). Os SRs abordam essa problemática filtrando esse grande volume de textos e dados, e recomendando as informações consideradas de maior relevância em conformidade a um perfil de interesse do leitor.

Sistemas recomendadores já são comuns em diversas áreas de produtos e serviços. A recomendação de notícias, entretanto, difere do cenário de recomendação de outros produtos em alguns aspectos. Primeiro, os artigos de notícias têm um ciclo de vida curto, forçando o recomendador a considerar continuamente novos artigos e descartar aqueles desatualizados. Segundo, os leitores estão interessados em vários tópicos e domínios variáveis, tornando difícil prever os artigos relevantes em um novo domínio. Um evento inesperado pode ser relevante para um leitor, embora nenhum evento semelhante tenha sido observado no passado. Por fim, um outro aspecto importante dos Sistema de Recomendação de Notícias (SRN) é que lidam com dados não estruturados que podem ser difíceis de interpretar, inconsistentes e incompletos (notícia sem tópico definido, por exemplo).

A estrutura geral de um Sistema Recomendador de Notícias (SRN) é aqui entendida como no diagrama de blocos da Figura 1.

**Figura 1 – Diagrama em blocos de um Sistema Recomendador de Notícias (SRN)**




Fonte: Elaborado pelo autor.

A funcionalidade do primeiro bloco é receber um fluxo de notícias de diversas Fontes de Notícias (FN), aplicar um pré-processamento apropriado e armazenar para uso posterior. Notícias são textos não estruturado em linguagem e estilo diversificado e precisam inicialmente de serem pré-processadas para armazenamento, recuperação e processamento eficientes.

A classificação de documentos de texto em uma ou mais de um conjunto de categorias (tópicos) pré-definidas é uma tarefa nuclear (*core*) do SRN. Entretanto, para ser eficaz nesta tarefa, a funcionalidade do segundo bloco da Figura 1 necessita de obter uma representação dos documentos adequada a esta função. Além de construir um SRN funcional, a funcionalidade deste bloco é o principal foco desta pesquisa.

As tarefas de pré-processamento, representação e classificação de textos é um dos pipelines típicos em Processamento de Linguagem Natural (PLN) (NORVIG; RUSSELL, 2014).

O problema da recomendação é o problema de associar as propriedades dos itens, os assuntos das notícias, neste caso, e o perfil do interesse dos assinantes. Assim, ele envolve dois subproblemas. O primeiro é a classificação dos itens, que são as notícias, em assuntos ou tópicos, e o segundo é a aquisição do perfil dos assinantes e a consequente distribuição das notícias. Este trabalho não desenvolve o problema da aquisição automática do perfil de interesse dos assinantes. Essa funcionalidade é implementada com os usuários inscrevendo-se diretamente nos tópicos do seu interesse em um catálogo de tópicos.

Uma questão que pode parecer pertinente é a de que hoje todas as agências produtoras de notícias são organizadas em editorias e geralmente as notícias já saem rotuladas por assunto. Nesse contexto o que justificaria o projeto de um Recomendador de Notícias? A resposta para essa questão pode ser assim explicada: Primeiro, entre as geradoras de notícias não há uma ontologia comum e nem as agências estão organizadas da mesma forma. Segundo, editoria[1] não é o mesmo que assunto. Terceiro, geralmente essa rotulação é monorrótulo sendo que sabe-se que a verdadeira classificação é multirrótulo, de forma que um sistema de recomendação agregador de notícias de várias fontes geradoras precisa de ter seu próprio sistema de classificação.

O SRN desenvolvido neste trabalho foi denominado **Luppar News Recommender (*Luppar News-Rec*)**. Ele é parte integrante do projeto Luppar (SILVA, 2018), o qual é um projeto de pesquisa em Recuperação de Informação (RI) na Universidade Estadual do Ceará. No restante desta introdução o leitor encontrará os objetivos, a metodologia e a estrutura desta dissertação.

[1] Conjunto das seções de uma publicação que estão a cargo de um editor.

## 1.1 OBJETIVOS

O objetivo deste trabalho é desenvolver um Sistema Recomendador para distribuição de Notícias (SRN). Um sistema como este é composto por subsistemas sendo o subsistema de **classificação de notícias** o *core* do SRN. Dessa forma, nos objetivos abaixo a atividade de pesquisa concentra-se neste subsistema.

**Objetivo Geral**: Construir e avaliar um Sistema Recomendador de Notícias (SRN) utilizando representações *embedding* de documentos.

**Objetivos Específicos**:

a) Especificar o contexto da aplicação e desenvolver uma estrutura para o SRN.
b) Obter propostas de representação *embedding* de documentos, baseada no *Word2Vec* e *FastText*, e comparar seu desempenho com representações *baselines*.
c) Construir um subsistema classificador monorrótulo e multirrótulo de notícias e avaliar combinações de algoritmos de classificação e representações de documentos para selecionar a melhor proposta para o SRN.
d) Implementar o Sistema de Recomendação de Notícias com interface de usuário para teste e demonstração online, incluindo aquisição, pré-processamento, armazenamento, classificação e distribuição de notícias.

## 1.2 METODOLOGIA

O desenvolvimento deste trabalho seguiu aproximadamente os seguintes passos:

- Definição do problema, do escopo da pesquisa e realização da revisão bibliográfica pertinente.
- Aquisição de bases de dados e planejamento dos experimentos. Foi decidido utilizar tanto bases publicamente disponíveis em língua inglesa quanto bases de aquisição e rotulação próprias em língua inglesa e portuguesa.
- Desenvolvimento de uma representação de documentos texto baseada em *embedding* e avaliação de combinações de representação com classificadores para a seleção da combinação a ser utilizada no ***Luppar News-Rec*** e avaliação da representação proposta.
- Concepção, programação e teste integrado das funcionalidades do ***Luppar News-Rec*** incluindo uma interface de usuário para facilidade de teste.

- Teste integrado do ***Luppar News-Rec***. Foram realizados testes de validação *offline* e *online* e avaliados os resultados. Para os testes foi utilizado o método chamado de validação cruzada (*cross-validation*) do tipo *10-fold*. Para o denominado Teste *Online*: Avaliação por Tópico e por Assinante, simulou-se a inscrição de 40 assinantes com perfis de interesse fictícios, cujos tópicos foram aleatoriamente selecionados no catálogo de tópicos previamente definidos. O sistema foi posto em funcionamento *online* até coletar e pré-processar 231.600 notícias, onde 206.578 são notícias curtas em língua Inglesa de duas fontes diferentes (*Reuters* e *Inshorts*), além de 25.022 em língua Portuguesa de diversas fontes (por exemplo: G1 Notícias, *Reuters* Brasil, UOL Notícias, BBC Brasil, Estado de São Paulo dentre outros). Essas notícias foram rotuladas automaticamente, com base na rotulação aplicada pelo site, para avaliar o desempenho do SRN quanto à classificação e distribuição previamente realizadas.

## 1.3 ESTRUTURA DA DISSERTAÇÃO

Após essa introdução, este trabalho continua com a seguinte estrutura:

- O Capítulo 2 conceitua representações de documentos texto (incluindo a contribuição deste trabalho para a representação de notícias), classificação de texto e sistemas de recomendação;
- O Capítulo 3 apresenta os trabalhos relacionados ao uso de representações de documentos, com foco em abordagens *embeddings*, na recomendação de notícias;
- O Capítulo 4 desenvolve o projeto do Recomendador de Notícias (***Luppar News-Rec***), sua arquitetura e subsistemas;
- O Capítulo 5 apresenta a avaliação e seleção das melhores combinações classificador/representação (essas são implementadas no ***Luppar News-Rec***), em seguida os resultados dos testes de sistema, desempenho e validação do ***Luppar News-Rec***, seguido de discussão dos pontos relevantes; e
- O Capítulo 6 é a conclusão.

# 2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta a fundamentação teórica de métodos utilizados para criação de Representação de Documentos, assim como para Classificação de Texto, Algoritmos de Classificação (KNN, SVM, *Decision Tree* e *Random Forest*), técnicas de Avaliação da Classificação, além de uma contextualização sobre Sistemas de Recomendação.

## 2.1 REPRESENTAÇÕES DE DOCUMENTOS

O Processamento de Linguagem Natural (PLN) é a subárea da Inteligência Artificial que estuda a capacidade e as limitações de uma máquina em entender a linguagem dos seres humanos (NORVIG; RUSSELL, 2014). O objetivo do PLN é fornecer aos computadores a capacidade de entender e compor textos. "Entender" um texto significa reconhecer o contexto, fazer análise sintática, semântica, léxica e morfológica, criar resumos, extrair informação, interpretar os sentidos, analisar sentimentos e até aprender conceitos com os textos processados.

Mesmo com o avanço no relacionamento homem-máquina, a comunicação via linguagem natural continua sendo um desafio: como criar programas capazes de interpretar mensagens codificadas em linguagem natural e decifrá-las para a linguagem de máquina? Com o passar dos anos, houve muitas pesquisas e desenvolvimentos nos mais diversos ramos do processamento de linguagem natural, destacando-se a tradução automática, considerada pela maioria como o marco inicial na utilização dos computadores para o estudo das línguas naturais.

De modo geral, no PLN, buscam-se soluções para questões computacionais que requerem o tratamento computacional de uma ou mais línguas naturais, quer sejam escritas ou faladas. Mais precisamente, o PLN dedica-se a propor e desenvolver sistemas computacionais que têm a língua natural escrita como objeto primário. Para tanto, cientistas buscam fundamentos em várias disciplinas: Filosofia da Linguagem, Psicologia, Lógica, Inteligência Artificial, Matemática, Ciência da Computação, Linguística Computacional e Linguística.

Para modelar a língua e possibilitar que a máquina a entenda, são necessários pré-processamentos que abstraem e estruturam a língua, deixando apenas o que é informação relevante. Essas informações são mapeadas normalmente em um vetor de características, as chamadas representações de documentos (MEDIUM, 2017).

A representação de documento mais simples para um texto é feita por meio dos termos/palavras que o compõe. Essa abordagem é conhecida como *Bag-of-words (BoW)* e representa os documentos como vetores no espaço de palavras registrando a frequência dos

termos em cada documento. Outras abordagens que podem ser utilizadas para representação de textos incluem: *Phrases, Bag-of-words with n-gram, Bag-of-words and word position, Concept categories* e outros (KOSALA; BLOCKEEL, 2000).

O processo de construção de qualquer uma das representações pode ser dividido em duas etapas. A primeira, denominada **Extração**, que decompõe o texto em elementos correspondentes aos atributos definidos pela representação escolhida. Essa etapa de pré-processamento tem forte dependência do idioma em que foi escrito o texto e normalmente filtros como *stopword* e *stemming* são aplicados (WANG; WANG, 2005). Na segunda etapa, a **Seleção**, os atributos extraídos são mapeados em um vetor de características. Associado a cada atributo se estabelece um peso, e alguns dos mais populares são: *Document Frequency (DF), Term Frequency (TF)* e *Inverse Document Frequency (IDF)* (SEBASTIANI, 2002).

Atualmente, técnicas de *Embeddings* estão sendo bastante utilizadas. Nesse trabalho elas serão comparadas com as mais populares e tradicionais (BoW e TF-IDF), além de uma abordagem proposta baseada em *Embeddings* ponderada por IDF. A ideia de ponderar por IDF não é nova e já foi aplicada em outros contextos e outras representações. Espera-se que ela apresente bom desempenho no contexto de SRNs.

### 2.1.1 Pré-processamento (Extração)

O pré-processamento de documentos é uma etapa que pode ser dividida em algumas transformações textuais, são elas:

1. Análise léxica do texto com o objetivo de tratar dígitos, hifens, marcas de pontuação e a caixa das letras, ou seja, se estão em maiúsculas ou minúsculas;
2. Tokenização é utilizado para decompor o documento em termos que o compõe. Os delimitadores utilizados para tokenização geralmente são: o espaço em branco entre os termos, quebras de linhas, tabulações, e alguns caracteres especiais;
3. Eliminação de *stopwords* com o objetivo de filtrar palavras com baixo poder discriminativo para objetivos de recuperação. Envolve remover artigos, preposições ou qualquer outro elemento que não possua representatividade semântica no domínio do problema;
4. *Stemming* (stemização) que envolve a redução de palavras às suas raízes, ou seja, tem como objetivo remover afixos (i.e., prefixos e sufixos) e permitir a recuperação de documentos contendo variações sintáticas dos termos de consulta, como por exemplo associar as palavras computador, computação e computado como a mesma raiz: computa (BAEZA-

YATES; RIBEIRO-NETO, 2013);

5. Alternativamente ao *Stemming* pode ser utilizada *Lemmatization* (Lematização) que é o ato de representar as palavras através do infinitivo dos verbos e masculino singular dos substantivos e adjetivos (LUCCA; NUNES, 2002).

### 2.1.2 Representações de Documentos

As representações de documentos utilizadas nesse trabalho são as tradicionais BoW e TF-IDF, assim como as representações de documentos *Embeddings* - *Word2Vec* e *FastText*, apresentadas nessa seção. Na seção seguinte é apresentada uma abordagem ponderada por IDF (`E2V-IDF`) para representações *Embeddings*.

***Bag-of-Words* (BoW)** - Um método comum no processamento de texto em linguagem natural é o modelo *Bag-of-words*, que representa um texto na forma de um saco de palavras, ou seja, levando em conta apenas a frequência das palavras no texto, sem se levar em consideração a ordem relativa entre elas ou a gramática (ZHANG; JIN; ZHOU, 2010). O modelo é frequentemente utilizado na classificação de documentos, e uma aplicação do modelo em recomendação de notícias pode ser encontra em (OH *et al.*, 2014). O exemplo a seguir mostra, de maneira simples, o resultado do algoritmo. Consideram-se três documentos distintos:

*1. I love dogs.*

*2. I hate dogs and knitting.*

*3. Knitting is my hobby and my passion.*

Com base nos documentos, monta-se o vetor de características, que é basicamente uma lista de todas as palavras contidas em ambos os documentos, sem repetição.

**Figura 2 – Visualização tabular do *Bag-of-Words (BoW)***

| | i | love | dogs | hate | and | knitting | is | my | hobby | passion |
|---|---|---|---|---|---|---|---|---|---|---|
| Doc 1 | 1 | 1 | 1 | | | | | | | |
| Doc 2 | 1 | | 1 | 1 | 1 | 1 | | | | |
| Doc 3 | | | | | 1 | 1 | 1 | 2 | 1 | 1 |

Fonte: Elaborado pelo autor.

Cada entrada do vetor refere-se à quantidade de vezes que aquela palavra aparece no documento em questão. Por exemplo, na Figura 2, o primeiro vetor corresponde ao primeiro documento, onde a palavra *"dogs"* aparece uma vez, portanto sua entrada tem o valor 1, e no

terceiro vetor, que corresponde ao terceiro documento, a palavra *"my"* aparece duas vezes e sua entrada tem o valor 2.

Um caso especial do *Bag-of-words* é quando as entradas do vetor são compostas de zeros e uns indicando apenas a presença ou ausência do termo no documento sem contar sua frequência. Esta representação é utilizada no modelo Booleano de recuperação de informação.

***Term Frequency-Inverse Document Frequency*** **(TF-IDF)** - É uma medida estatística que representa a importância de uma palavra (ou "termo") para a individualização de um documento. Ela aumenta quando essa palavra aparece muitas vezes nesse documento, e diminui caso ela seja frequente em muitos outros documentos da coleção. Seu cálculo consiste na multiplicação da frequência da palavra no documento (TF) pelo inverso de sua frequência nos documentos da coleção (IDF), tendo diversas variações no cálculo de cada um desses fatores, como mostrado em (KOMPAN; BIELIKOVÁ, 2010).

O TF de uma palavra dentro de um documento é usualmente calculada como o número de vezes que ela aparece nesse documento dividido pelo número total de palavras, ou seja, é a frequência relativa da palavra no documento. Já o IDF, representando a importância de uma palavra, tem um valor menor para as palavras frequentes em todos os documentos e valor maior para as palavras mais raras. Assim, os artigos e as preposições, que geralmente ocupam a maior parcela de uma sentença, tem valores reduzidos (BAEZA-YATES; RIBEIRO-NETO, 2013).

A forma mais comum de representar essa medida, usada neste trabalho, é usando a equação:

$$\text{tf-idf}_{t,d} = (1 + log\ \text{tf}_{t,d}).log\frac{N}{\text{df}_t} \tag{2.1}$$

onde $t$ é o termo, $d$ é o documento, $\text{tf}_{t,d}$ é a frequência do termo $t$ no documento $d$, $N$ é o número total de documentos e $\text{df}_t$, o número de documentos que contém o termo $t$.

A Figura 3, demonstra os três vetores expostos acima, que representa três documentos, agora utilizando o TF-IDF. Podemos observar que, um dos problemas do TF-IDF é que quanto maior o número de documentos e termos, maior a dimensão da matriz gerada (em grande volumes de dados, temos uma alta dimensionalidade), além de ser uma matriz esparsa. Segundo (PISSANETZKY, 1984) existem muitas definições de matriz esparsa na literatura, entretanto

todas possuem em comum o conceito de que uma matriz é considerada esparsa se muitos de seus elementos são iguais a zero (nas Figuras 2 e 3, os valores zero estão como vazio).

**Figura 3 – Visualização tabular do *Term Frequency-Inverse Document Frequency* (TF-IDF)**

| | i | love | dogs | hate | and | knitting | is | my | hobby | passion |
|---|---|---|---|---|---|---|---|---|---|---|
| Doc 1 | 0.18 | 0.48 | 0.18 | | | | | | | |
| Doc 2 | 0.18 | | 0.18 | 0.48 | 0.18 | 0.18 | | | | |
| Doc 3 | | | | | 0.18 | 0.18 | 0.48 | 0.95 | 0.48 | 0.48 |

Fonte: Elaborado pelo autor.

Outro problema dessa abordagem é a ambiguidade semântica, que ocorre por conta da polissemia da palavra ou da expressão. Segundo a gramática de (PASCHOALIN; SPADOTO, 1996), "Polissemia é o fato de a palavra poder apresentar significados diferentes que se explicam dentro do contexto". Por exemplo, na frase "Espero você no banco", a palavra banco pode se referir a uma agência bancária ou um banco de praça.

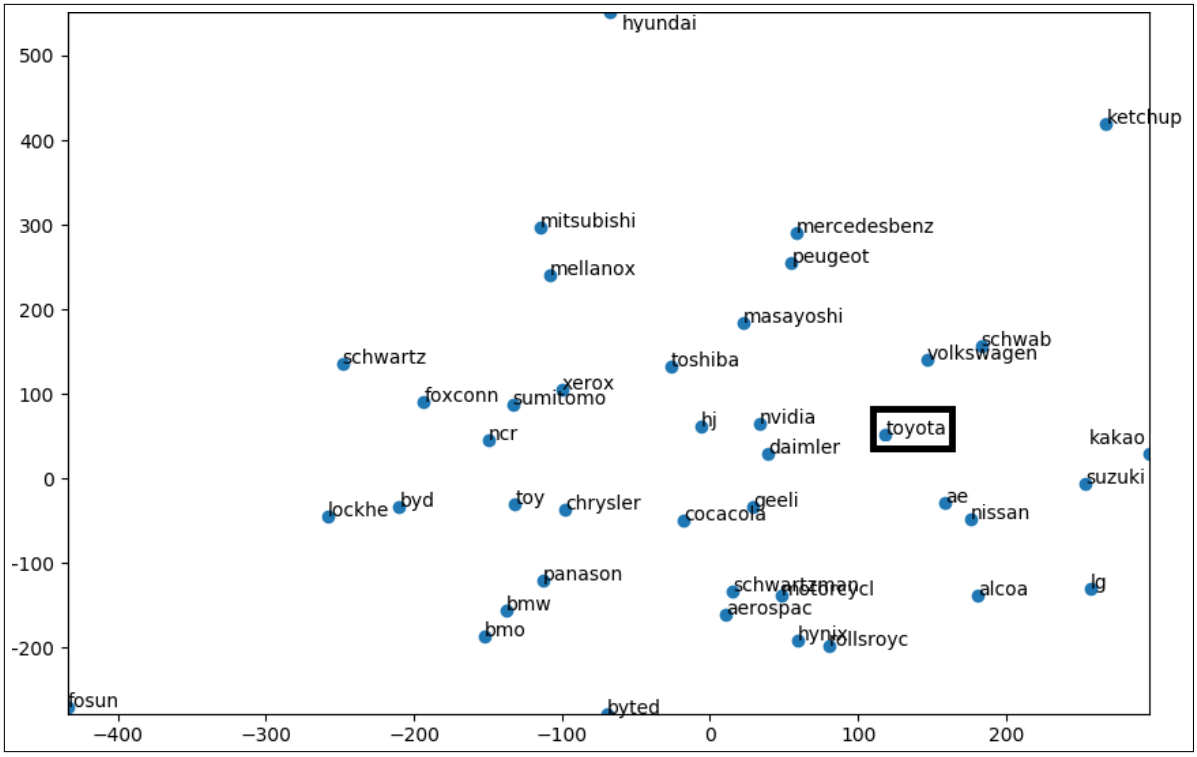
***Word Embeddings*** - A representação por *Word Embedding* é uma das formas de amenizar os problemas encontrados em modelos BoW e TF-IDF, como a alta dimensionalidade e ambiguidade semântica (polissemia e sinonímia, o primeiro consiste em uma palavra que possui vários significados, o segundo são palavras diferentes que possuem o mesmo significado). Neste trabalho foram avaliadas as representações *Word2Vec* e *FastText*, ambas trabalham com modelos de semântica distribucional, que serão detalhadas a seguir.

O *Word2Vec*, proposto por (MIKOLOV *et al.*, 2013) e aplicado por (MOREIRA; FERREIRA; CUNHA, 2018), é uma técnica que se propõem a representar uma palavra por um vetor de números reais, denso e de tamanho arbitrário pré-definido, utilizando redes neurais como modelos de linguagens. Os resultados obtidos por (MIKOLOV *et al.*, 2013) mostram que a representação vetorial é capaz de capturar parte do sentido semântico dos termos. Na prática, isso significa que a representação de palavras sinônimas, por exemplo, ficam próximas entre si no espaço de vetores *embedding* obtido. Na Figura 4, podemos ver as palavras vizinhas à palavra Toyota em um vetor *embedding* de duas dimensões, e observamos que outras marcas, não apenas marcas de veículos, estão localizadas bem próximas uma das outras. Segundo (SOUSA, 2016), esse método tem como base aprender uma representação para cada palavra, chamada de *embeddings*, as palavras são mapeadas para vetores exemplo "vec(cão) = (0.0, 1.6, -1)". O

*Word Embedding* aprende muitas regularidades e padrões linguísticos. Seja "vec(palavra)" a representação de uma dada palavra. A seguinte operação vetorial: *vec(Madri)* - *vec(Espanha)* + *vec(França)* = *vec(Paris)*, apresenta um caso de relação que pode ser aprendida.

No *FastText*, proposto por (BOJANOWSKI *et al.*, 2017) e utilizado por (SAMARINAS; ZAFEIRIOU, 2019), os *embeddings* são associados com n-grams de caracteres e as palavras são representadas como a composição dessa representação de caracteres. Este método tenta capturar e adicionar nos *word embeddings* informação morfológica das palavras.

**Figura 4 – *Embedding* - *Word2Vec* (Palavra: Toyota) - Utilizando *t-Distributed Stochastic Neighbor Embedding* (t-SNE), uma técnica para redução de dimensionalidade que é particularmente adequada para a visualização de conjuntos de dados de alta dimensão**




Fonte: Elaborado pelo autor.

Foram desenvolvidos, por (MIKOLOV *et al.*, 2013), dois tipos de modelos de semântica distribucional[1]: o *Skip-gram* e *Continuos Bag-of-words (CBOW)*, eles se encontram na categoria de modelos preditivos[2], também chamados de modelos neurais de linguagem, que tentam predizer uma palavra ou mais, por meio de uma rede neural simplificada, dado como

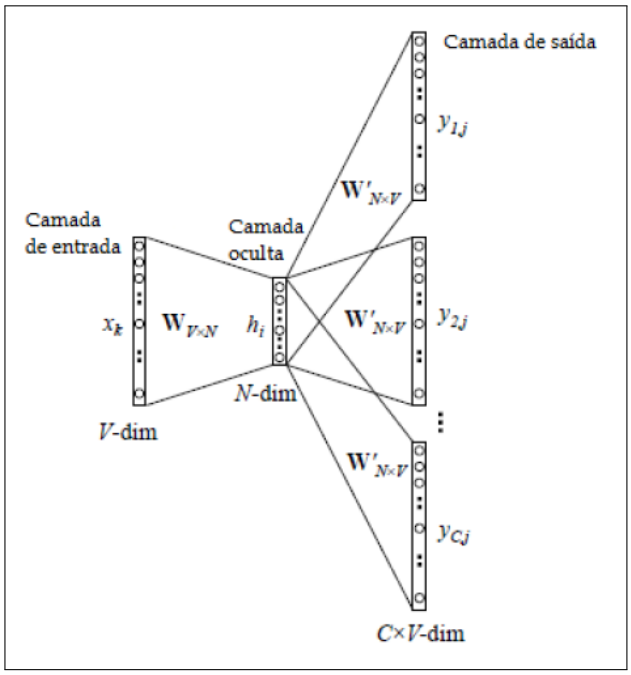
[1] São modelos que baseiam se pressuposto que o significado de uma palavra está relacionado com a sua utilização no cotidiano ou de sua ditribuição no texto (SOUSA, 2016)

[2] Modelos que são treinados utilizando redes neurais artificiais (SOUSA, 2016)

entrada uma palavra ou um conjunto de palavras.

O *Skip-gram* é um modelo que aprende vetores de representação de palavra utilizando redes neurais. O cálculo empregado codifica padrões linguísticos nos vetores gerados. Como exemplo, a analogia "rei está para rainha assim como homem está para mulher" é codificado em um espaço vetorial no qual a equação: *vec("rei") - vec("rainha") = vec("homem") - vec("mulher").*

**Figura 5 – Arquitetura do modelo *Skip-gram***




Fonte: SOUSA (2016).

O modelo *Skip-gram* busca, por meio de geração de um vetor de representação de palavras, predizer as palavras vizinhas que se encontram ao redor de uma determinada palavra. O modelo *Skip-gram* foi introduzido por (MIKOLOV *et al.*, 2013) que realça em seu estudo a baixa complexidade computacional que seus modelos possuem, e como isso aprimora o tempo de treinamento em um *corpus* (coleção de documentos) grande.

Conforme a Figura 5 o modelo *Skip-gram* possui duas matrizes a $W_{|V|xN}$ que conecta a camada de entrada a camada oculta,sendo $|V|$ o tamanho do vocabulário (termos da coleção) e $N$ o tamanho de dimensões, e a matriz $W'_{Nx|V|}$, que liga a camada oculta a camada de saída. Para o desenvolvimento do método, primeiro tem se a entrada da palavra em representação *One-Hot*[3], que será utilizada para prever o contexto, em seguida multiplica-se a matriz $W$ pelo vetor da

[3] *One-Hot* consiste em tratar as palavras como símbolos atômicos, ou seja uma representação discreta, na qual cada palavra é um atributo diferente

palavra $w_t$ , obtendo um vetor $u_{1xN}$ (este vetor é o *embedding* usado na entrada da rede). A camada oculta denominada $h$ recebe o vetor gerado pela multiplicação da matriz $W$ com o vetor da palavra $w_t(h = u_{1xN})$.

Para cada uma das palavras contextos, multiplica-se o vetor $h$ pela matriz $W'$ (que também armazena *embeddings*, mas usados na camada de saída), como resultado do cálculo de cada uma das palavras contexto, $v = hW'$, aplica-se a função *softmax*[4] nos vetores $v$ gerados, dessa forma obtém-se para cada palavra contexto um vetor $1x|V|$, com pontuações para cada uma das palavras do vocabulário. A seguir, o vetor *one-hot* da saída pode ser recuperado se desejável (SOUSA, 2016).

O modelo *CBOW* tem como objetivo prever ou representar uma certa palavra baseando-se no contexto em que ela está inserida, exemplo: “o” “gato” “na” “mesa” é tido como contexto de entrada, e a palavra “pulou” seria provavelmente o alvo a ser alcançado. O modelo apresenta maior simplicidade em relação ao modelo anterior, o seu diferencial em relação ao modelo *Bag-of-Words* (BoW) é a técnica de representação de palavras que utiliza representação distribuída (MIKOLOV *et al.*, 2013).

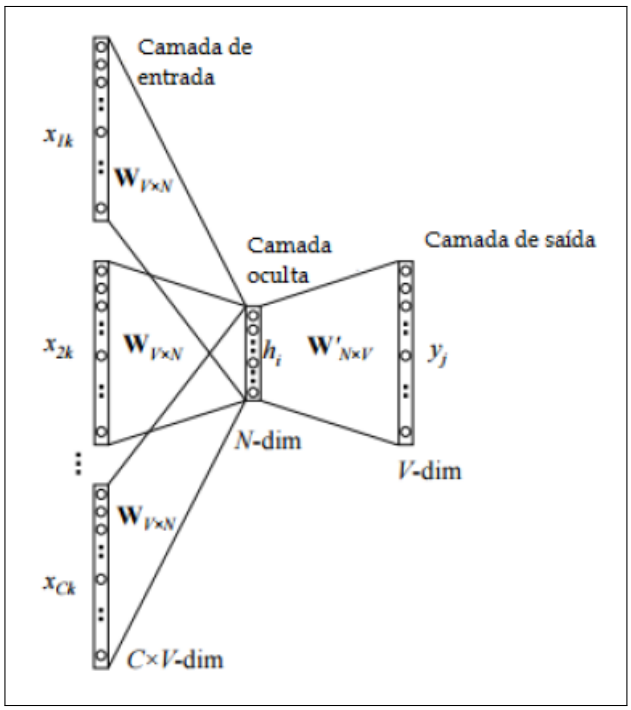
O CBOW trabalha com a predição de palavras assim como o *Skip-gram*, a diferença entre eles é que o CBOW prediz uma palavra, conforme a janela de palavras contexto ao redor, enquanto o *Skip-gram* prediz o contexto a partir da palavra. O modelo tenta predizer uma palavra $w_t$ de uma janela de treinamento.

Na Figura 6 acima, está a arquitetura do método CBOW que é composta pela camada de entrada, camada oculta e camada de saída. A camada de entrada é conectada a camada oculta por uma matriz de contexto $W_{|V|xN}$, onde o número de colunas $N$ é um valor arbitrário do tamanho de características, e suas linhas $|V|$ representam os vetores das palavras do vocabulário. Na camada de entrada, as palavras da janela que serão utilizadas para predizer $w_j$, são representadas em vetores *One-Hot*, os vetores são multiplicados pela matriz $W$, em seguida é realizada a média dos vetores *embeddings* decorrentes da multiplicação, resultando em um vetor somente, obtendo dessa forma a camada oculta.

O vetor da camada oculta $h$ é multiplicado pela matriz $W'_{Nx|V|}$ para se obter o vetor de saída $z$, que após a aplicação da função *softmax*, gera um vetor $y$ $1x|V|$, que representa uma

[4] A função de ativação *softmax* é usada em redes neurais de classificação. Ela força a saída de uma rede neural a representar a probabilidade dos dados serem de uma das classes definidas

**Figura 6 – Arquitetura do modelo *CBOW***




Fonte: SOUSA (2016).

pontuação para cada palavra do $|V|$ (SOUSA, 2016). A palavra que tiver maior escore será a predita.

Nos modelos *Skip-gram* e CBOW, o cálculo de probabilidade utilizado, conhecido como *softmax* comentado anteriormente, apresenta um custo computacional caro, por estar diretamente ligado ao tamanho do vocabulário, que pode chegar na casa dos milhões, sendo impraticável nestas situações. Nesse trabalho foi utilizado em substituição ao *softmax* o Amostras Negativas, método que busca diminuir a complexidade do cálculo de probabilidade dos modelos, por meio da redução de vetores que devem ser atualizados a cada iteração de treinamento. Essa nova abordagem proposta por (MIKOLOV *et al.*, 2013) chamada *Amostras Negativas* demonstrou eficiência na criação de *Word Embeddings*.

Amostras Negativas são uma simplificação do modelo *Noise Contrastive Estimation* (NCE) que mantém a qualidade de representação dos vetores. O NCE expõe que um bom modelo deve saber diferenciar dados de ruídos. No modelo são geradas amostras positivas e negativas, pares de palavras e contextos. As positivas são extraídas do *corpus*, enquanto que as negativas são definidas utilizando uma distribuição ruído que sorteia palavras e contextos aleatórios, com maior probabilidade para palavras frequentes do vocabulário. A tarefa do modelo é desenvolver uma função objetivo que maximize a probabilidade de um par de palavra e contexto estar no *corpus*, se realmente estiverem, e maximizar a probabilidade de um par não estar no *corpus*, se

eles forem sorteados aleatóriamente (SOUSA, 2016).

Representação *Embedding* de documentos, a partir da representação *embedding* das palavras é necessário obter uma representação de documentos. Para isso, foram avaliadas duas abordagens. A primeira é uma representação na qual um documento (notícia) é representado pelo vetor média dos vetores das palavras que o compõe, utilizada em (KIM; KIM; CHO, 2017) e (MARTINČIĆ-IPŠIĆ; MILIČIĆ; TODOROVSKI, 2019). Essa técnica frequentemente utilizada será tomada com uma das *baselines* deste trabalho. A segunda é uma abordagem de média ponderada apresentada neste trabalho, que será descrita a seguir.

Como um exemplo demonstrativo, reescrevendo os três vetores da Figura 2, que representam três documentos, agora utilizando a abordagem *Embedding - Word2Vec/FastText* (300 dimensões - detalhes da escolha na seção 5.1.1), obteremos algo como mostra a Figura 7:

**Figura 7 – Visualização tabular de uma representação *Embedding* de documentos**

| | - | - | - | - | - | - | - | - | ... | - |
|---|---|---|---|---|---|---|---|---|---|---|
| Doc 1 | -0.82 | -0.04 | 0.67 | 0.35 | -0.30 | -0.66 | 1.25 | -1.11 | ... | 1.63 |
| Doc 2 | 1.32 | 0.98 | -0.10 | -1.48 | 0.23 | -0.71 | -0.87 | -0.18 | ... | -0.89 |
| Doc 3 | -0.51 | 1.54 | -0.56 | 0.67 | -0.32 | -0.42 | 1.99 | 0.91 | ... | -1.21 |

Fonte: Elaborado pelo autor.

### 2.1.3 A abordagem `E2V-IDF`

Uma abordagem comum utilizada para ir da representação *embedding* das palavras para uma representação de documentos é a representação na qual um documento (notícia) é representado pela média dos vetores dos termos que o compõe (KIM; KIM; CHO, 2017).

A abordagem deste trabalho, ao invés, representa um documento pela média dos vetores dos seus termos, ponderando cada vetor de termo pelo IDF (Inverso da Frequência nos Documentos) do termo. A intuição por trás desta proposta é que um termo, apresente poder discriminatório diferente dependendo do número de documentos em que esse termo esteja presente, ou seja, o peso dos termos que ocorrem com mais frequência em documentos da coleção tendem a diminuir, e aumentar caso os termos ocorram mais raramente em documentos da coleção (SOUZA; MAIA, 2019).

Essa abordagem será identificada pelo acrônimo `E2V-IDF` (*Embeddings* ponderado por IDF) e implementada no ***Luppar News-Rec***. A Figura 8 apresenta o pseudocódigo para obter a representação de documentos `E2V-IDF`. A abordagem aceita como entrada vetores *embeddings*

tais como *Word2Vec, FastText, Glove* e *Wang2Vec*[5].

**Figura 8 – Pseudocódigo da Representação de Documento** `E2V-IDF`

```
Data: DOCUMENTOS (Notícias), VETOR EMBEDDING (Vetor de cada termo da
      coleção)
Result: Vetor E2V-IDF de cada Documento
while Existir Termos na Coleção do
    Calcula_IDF_Termo()    //Monta o vetor: wIDF (Termo e IDF do Termo);
end
while Existir Documentos na Coleção do
    while Existir Termos no Documento do
        E2V-IDF(documento) = Média (VETOR EMBEDDING(termo) * wIDF(Termo));
    end
end
```

Fonte: Elaborado pelo autor.

A ponderação por termos já foi vista antes em (BARUAH; KOLLA, 2018) onde os autores utilizam o *Word2Vec* ponderado por IDF. Porém de forma diferente, onde o IDF do termo é obtido por uma coleção de documentos diferente da coleção de documento de treinamento utilizada no *CL-SciSumm 2018*. A coleção de documentos utilizada para encontrar o IDF do termo é a *Parcit Structured XML*, uma versão do *ACL Anthology Reference Corpus*, já a coleção de treinamento utilizada no *CL-SciSumm 2018* é a *Training Set for 2018* que foi criada pela amostragem aleatória de dez documentos do corpus *ACL Anthology Reference Corpus* e pela seleção de seus documentos de citação. Concluíram que os pesos IDF de termos podem aumentar a eficácia dos métodos baseados na média de *Embeddings (Word2Vec*) de palavras para similaridade de sentenças.

Já em (REI; CUMMINS, 2016) os autores utilizam *Embeddings (Word2Vec)* ponderado por *IDF* para representar sentenças, entretanto de forma diferente da abordagem aqui estudada, na qual cada palavra recebe um peso separado dado por $IDF(w) = log(N/1+n_w)$, onde $w$ é a palavra, $N$ é o numero total de sentenças da coleção e $n_w$ é o número de sentenças em que a palavra-alvo $w$ ocorre.

Por outro lado, nesse trabalho, a ponderação *IDF* é dada pela fórmula:

$$IDF(w) = log\left(\frac{N}{df_t}\right) + 1 \tag{2.2}$$

[5] Glove e Wang2Vec - disponíveis em http://nilc.icmc.usp.br/embeddings

onde $w$ é o termo (palavra), $N$ é o número total de documentos da coleção e $df_t$ é o número de documentos em que o termo $w$ ocorre. A composição $log() + 1$ em vez de $log$ garante que os termos com *IDF* igual a zero não sejam suprimidos inteiramente. Além disso, para $df_t$ ou $n_w$ grandes as duas fórmulas dão os mesmos resultados entretanto para valores pequenos elas se diferenciam bastante.

## 2.2 CLASSIFICAÇÃO DE TEXTOS

Desde os tempos antigos, dos primeiros dias da Grande Biblioteca de Alexandria, por volta de 300 a.C., bibliotecários tinham que lidar com o armazenamento de documentos para recuperação e leitura futura. Com o passar do tempo, o tamanho das coleções cresceram e o problema tornou-se cada vez mais difícil. Procurar por um livro em particular dentre centenas de livros tornou-se uma tarefa tediosa, demorada e impraticável. Para aliviar o problema, bibliotecários começaram a rotular os documentos. Isso forneceu metainformação ao seu conteúdo, permitindo assim que os livros fossem organizados com uma visão que permitia busca rápida e recuperação.

Uma das primeiras abordagens para rotular documentos foi atribuir um identificador único para cada documento. Isso resolvia o problema sempre que o usuário soubesse dos identificadores dos livros que eles queriam, mas não resolvia o problema mais genérico de encontrar documentos sobre um assunto ou tópico específico. Nesse caso, a solução natural é agrupar os documentos por tópicos comuns e nomear cada grupo com um ou mais rótulos significativos.

Cada grupo rotulado é o que chamamos de uma classe, isto é, um conjunto no qual podemos inserir documentos cujo conteúdo pode ser descrito pelo mesmo rótulo. Para ilustrar, podemos criar uma classe rotulada "cirurgia de miocárdio" e inserir nesta classe todos os documentos que descrevem procedimentos cirúrgicos para tratar de doenças do miocárdio. Nesse caso, o rótulo da classe descreve um tópico referenciado pelos documentos e, por causa disso, a tarefa de classificação é comumente referida como classificação de tópicos. Essa é provavelmente a variação mais importante do problema de classificação e, portanto, nossa discussão e nossos exemplos são baseados nela.

Classes podem ser usadas para descrever não somente tópicos, mas também outras características dos documentos, como linguagem, gênero, qualidade, autoridade e popularidade. Para exemplificar, considere um conjunto de críticas de restaurantes e assuma que necessitamos

distinguir entre críticas boas e ruins. Tal problema pode ser modelado como um problema de classificação binária com duas classes: críticas boas e críticas ruins. Nesse caso, as classes referem-se à qualidade dos documentos como críticas, e não a tópicos diretamente referenciados por eles mas se as críticas são positivas ou negativas.

O processo de inserção dos documentos em classes, i.e., de associação de um ou mais rótulos de classe em cada documento, é comumente chamado de classificação de textos. Algumas vezes, as classes são chamadas de categorias e, nesse caso, o processo é dito ser de categorização de textos. Aqui, consideramos que a classificação e a categorização são o mesmo processo e não faremos distinção entre elas (BAEZA-YATES; RIBEIRO-NETO, 2013).

Nesta seção apresentam-se os algoritmos clássicos para a classificação de textos utilizados no trabalho.

### 2.2.1 Uma caracterização da classificação de textos

Muitos dos algoritmos de classificação de textos foram desenvolvidos por pesquisadores que trabalhavam na área de aprendizagem de máquina. Assim sendo, antes de prosseguir, discute-se o aprendizado de máquina e sua relação com a classificação de textos.

**Aprendizado de Máquina** - O aprendizado de máquina é uma área ampla da Inteligência Artificial preocupada com o projeto e o desenvolvimento de algoritmos que aprendem padrões presentes nos dados fornecidos como entrada. Os padrões aprendidos são então usados para fazer predições relativas a dados ainda não vistos e novos. Aplicações de algoritmos de aprendizado de máquina incluem processamento da língua natural, diagnóstico médico, detecção de fraudes em cartões de crédito, análise de mercado de ações, visão computacional, Recuperação de Informação, e outros.

Algoritmos de aprendizado de máquina são dependentes de uma fase de aprendizado, a qual é usada para produzir um modelo ou uma função que codifica padrões presentes nos dados de entrada. Dependendo do mecanismo de aprendizado usado, os algoritmos de aprendizado de máquina podem ser basicamente de três tipos: aprendizado não supervisionado, aprendizado semi-supervisionado e aprendizado supervisionado. Outros tipos de algoritmos de aprendizado incluem o aprendizado por reforço e a transdução, mas eles não serão discutidos aqui.

O aprendizado não supervisionado (RUSSELL; NORVIG, 2016) distingue-se do aprendizado supervisionado no sentido em que nenhuma informação de classe no dado de treinamento é fornecida. Os algoritmos de aprendizado não supervisionado incluem modelos de redes

neurais, análise de componentes independentes e *clustering*. O aprendizado semi-supervisionado combina uma pequena quantidade de dados rotulados com uma quantidade maior de dados não rotulados para melhorar as predições. Além disso, as predições melhoradas são produzidas sem que seja necessário realizar a demorada atividade de rotulagem de grandes quantidades de dados. Apesar de não discutir aqui mais a fundo o aprendizado semi-supervisionado, o leitor interessado é convidado a consultar (ZHU, 2005) para uma revisão da literatura na área.

O aprendizado supervisionado requer o aprendizado de uma função a partir dos dados de treinamento fornecidos como entrada. No caso da classificação de textos, os dados de treinamento são compostos de pares do tipo documento-classe indicando as classes corretas para os documentos dados, geralmente de acordo com especialistas humanos. Esses dados de treinamento são então usados para aprender uma função de classificação, a qual pode ser usada para fazer predições de classes para dados ainda não vistos ou novos. A abordagem só funciona se a função aprendida for tal que ela possa pegar dados que não foram vistos antes e prever classes para eles com alta precisão. Os algoritmos de classificação utilizados neste trabalho são supervisionados (BAEZA-YATES; RIBEIRO-NETO, 2013).

**O problema da classificação de texto** - a classificação de texto pode ser dividida em classificação monorrótulo, no qual os exemplos a serem classificados estão associados a apenas um único rótulo. Quando o conjunto de rótulos contém apenas dois rótulos possíveis para ser associado a um exemplo, temos a classificação monorrótulo binária e quando esse número de rótulos é maior que dois, temos a classificação monorrótulo multiclasse, aqui chamada apenas de **classificação monorrótulo**. Já a **classificação multirrótulo** difere de classificação de rótulo único porque um exemplo pode está associado a mais de um rótulo e pela possibilidade de considerar a correlação entre rótulos como fonte de informação na classificação.

Para uma descrição formal do problema de classificação, seja $X = (x_1, ..., x_n)$ um conjunto finito de instâncias (documentos, notícias) e $Y = (y_1, ..., y_m)$ um conjunto finito de rótulos (tópicos), onde dado um conjunto de exemplos de treinamento $S = \{(x_1, Y_1), ..., (x_n, Y_m)\}$ a tarefa de aprendizagem de máquina é construir um classificador $f : X \to Y$, onde de acordo com a cardinalidade do conjunto $Y$, o problema pode ser de classificação monorrótulo binária (quando $|Y| = 2$) ou monorrótulo multiclasse (quando $|Y| > 2$). Já no caso da classificação multirrótulo o objetivo é gerar um mapeamento de $X$ para um subconjunto de rótulos de $Y$, ou seja o classificador $f : X \to 2^Y$ capaz de estimar uma função alvo desconhecida $\varphi : X \to 2^Y$. Neste contexto, o conjunto potência $2^Y$ representa todos os possíveis perfis de assinante ou todos

possíveis rótulos para uma notícia.

Os classificadores geralmente geram escores para cada uma das classes quando se aplica um novo exemplo. Para classificação monorótulo, em geral, toma-se a classe de maior escore. Para classificação multirótulo toma-se os N maiores escores quando se deseja classificar em N rótulos.

### 2.2.2 Algoritmos de classificação de textos

Algoritmos de Classificação são usados quando é necessário associar um documento em classes pré-estabelecidas. Um usuário, à medida que fornece ao sistema mais indícios de suas preferências, possibilita um maior aprimoramento das recomendações baseadas em conteúdo textual. Neste processo de avaliar se um documento será interessante ou não para o usuário, os algoritmos de classificação são responsáveis por predizer a qual ou quais classes de interesse pertence o item.

Entre os diversos algoritmos encontrado na literatura, os seguintes foram escolhidos para serem avaliados nesta tarefa de classificação de notícias. Essa escolha se realizou com base em trabalhos anteriores que os utilizou em tarefas de PLN. Note-se que são todos classificadores não lineares que normalmente apresentam bom desempenho em tarefas de classificação com dados não linearmente separáveis.

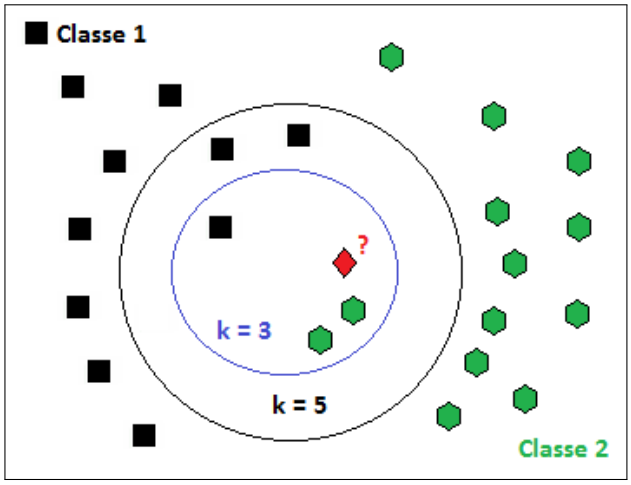
- Classificador KNN (*K-Nearest Neighbor*), utilizados em (PAWAR; GAWANDE, 2012) e (THAOROIJAM, 2014);
- Classificador SVM (*Support Vector Machines*), estudado nos trabalhos (PAWAR; GAWANDE, 2012), (KAUR; KAUR, 2017) e (SHAHI; PANT, 2018);
- Classificador DT (*Decision Tree*), utilizado em (PAWAR; GAWANDE, 2012) e (THAOROIJAM, 2014);
- Classificador RF (*Random Forest*), avaliado em (TELOKEN *et al.*, 2016) e (MARTINČIĆ-IPŠIĆ; MILIČIĆ; TODOROVSKI, 2019).

Esses serão os algoritmos utilizados nas tarefas de classificação nesse trabalho e detalhados a seguir.

***KNN* (*K-Nearest Neighbor*)**: Classifica novas amostras de acordo com as $K$ amostras do conjunto de treinamento mais próximas a essas novas amostras, como pode-se ver na Figura 9.

*KNN* usa uma medida de distância para definir a proximidade de uma amostra com outra (DUDA; HART; STORK, 1995). Dado um conjunto de $n$ pares $\{(x_1,\theta_1),...,(x_n,\theta_n)\}$, em que $x_i$ toma valores de um espaço $X$, e $\theta_i$ toma valores de um conjunto $1,2,...,M$ de classes. Considera-se cada $\theta_i$ como o índice do tópico a que pertence o *i-ésimo* indivíduo, e cada $x_i$ o resultado de um conjunto de medições feitas sobre aquele indivíduo. Se é dado um novo par $(x,\theta)$, no qual apenas o valor de $x$ é conhecido e deseja-se estimar $\theta$ a partir do conjunto de pontos classificados corretamente, dize-se que $x_n^{'} \in \{x_1,x_2,...,x_n\}$ é o vizinho mais próximo de $x$ se $min\ d(x_i,x) = d(x_n^{'},x)$, com $i = 1,2,...,n$. A regra vizinho mais próximo decide que $x$ pertence ao tópico $\theta_n^{'}$ de seu vizinho mais próximo $x_n^{'}$. A distância $d$ é determinada por uma métrica de similaridade, geralmente a distância Euclidiana (utilizada no trabalho, com valor de $K$ = 5).

**Figura 9 – Classificação de uma nova amostra através do método KNN, a figura apresenta dois valores para K (K=3 e K=5), para o K=3, é calculado a menor distância entre a nova amostra e seus três vizinhos mais próximos, podemos visualizar pela figura que a nova amostra será atribuída à Classe 2**

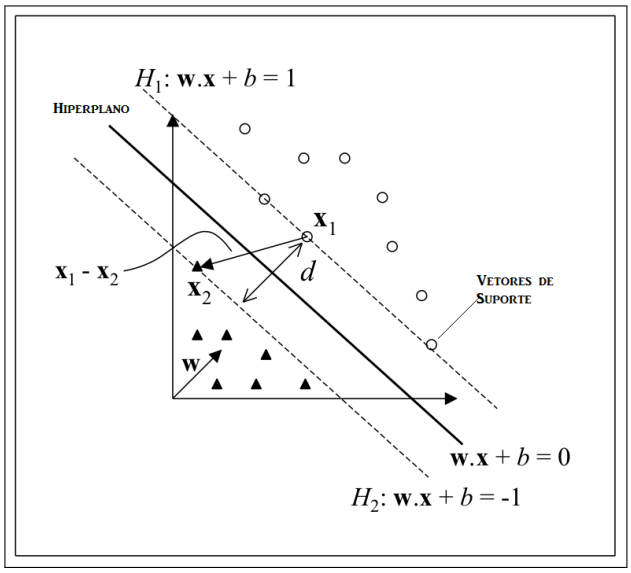



Fonte: Elaborado pelo autor.

Apesar de sua simplicidade, o *KNN* apresenta um bom desempenho, mas possui algumas desvantagens conhecidas (PAWAR; GAWANDE, 2012): a necessidade de armazenar o conjunto de treinamento; o alto custo computacional para calcular a distância entre a nova amostra e todas outras do conjunto de treinamento; baixa precisão em espaços de características muito elevados e dificuldade em se definir o melhor valor do parâmetro $K$.

***SVM* (*Support Vector Machines*)**: No *SVM*, a classificação se baseia na margem de separação das classes. Assim, o objetivo do treinamento do *SVM* consiste em encontrar um hiperplano separador ótimo, ou seja, aquele em que a distância de separação entre as classes é máxima ($d$), chamado hiperplano de margem máxima, ilustrado na Figura 10. As amostras que estão situadas sobre as margens são as mais informativas para a criação do limite de decisão da classificação e são chamadas de vetores suporte. O problema resultante da formulação SVM é um problema quadrático (SCHOLKOPF; SMOLA, 2001).

**Figura 10 – Cálculo da distância $d$ entre os hiperplanos H1 e H2**




Fonte: Adaptada do artigo de Lorena e Carvalho (2007).

A classificação pode ser realizada tanto no espaço original dos atributos quanto em um espaço de características projetado através de uma função de *kernel*. Assim, problemas que não são linearmente separáveis no espaço original podem tornar-se linearmente separáveis no espaço de características. À medida que a dimensão do espaço de características aumenta, também aumenta a probabilidade desse problema se tornar linearmente separável (SCHOLKOPF; SMOLA, 2001). A habilidade de separar dados com distribuição não linearmente separável depende da escolha da função *kernel* escolhida, e deve ser analisada de acordo com o domínio do problema estudado. Os *kernels* mais usados são: Linear, Polinomial e Função de Base Radial (*Radial Basis Function* - RBF). Neste trabalho foi utilizado o *kernel* RBF (escolha do *kernel* detalhada na seção: 5.1.1), como em (SHAHI; PANT, 2018).

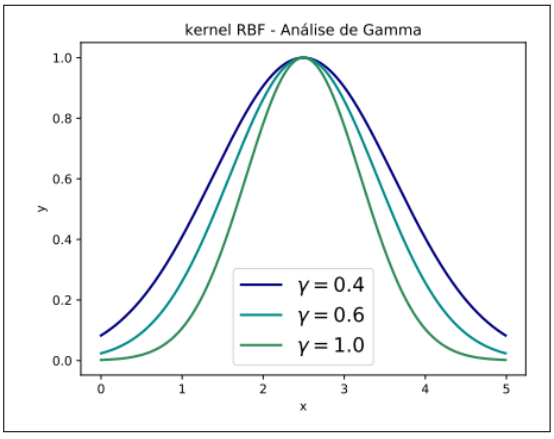
Este *kernel*, expresso pela Equação 2.3, possui um parâmetro de abertura $\gamma$ que deve ser escolhido de forma adequada.

$$K(x_i, x_j) = exp(\gamma||x_i - x_j||^2) \tag{2.3}$$

Conforme (FONSECA *et al.*, 2017), um *kernel* expressa uma medida de semelhança entre dois vetores. No caso do *kernel* RBF, esta representação de semelhança é feita através de uma função que decai de acordo com a distância entre os vetores. Deste modo, os vetores próximos são mais significativos na função *kernel* RBF quando comparados a vetores afastados. Esta função tem uma forma de Curva de Gauss ou Curva do Sino (*Bell Curve*). O parâmetro $\gamma$ da função *kernel* RBF define o decaimento (a largura da Curva de Sino). Quando o valor de $\gamma$ é pequeno, curvas largas são produzidas.

A Figura 11 apresenta uma análise do efeito do parâmetro $\gamma$ (*gamma*) na função *kernel* RBF (*Radial Basis Function*).

**Figura 11 – Análise do parâmetro $\gamma$ na função *kernel*: RBF**




Fonte: FONSECA (2017).

Segundo (LORENA; CARVALHO, 2007) os SVMs trabalham na obtenção de fronteiras lineares para a separação de dados pertencentes a duas classes. Podem ser divididos em problemas linearmente separáveis, o chamado: *SVM de Margens Rígidas* (BOSER; GUYON; VAPNIK, 1992), cuja formulação foi posteriormente estendida para definir fronteiras lineares sobre conjuntos de dados mais gerais, o: *SVM de Margens Suaves* - detalhado a seguir.

Sendo $T$ é um conjunto de treinamento com $n$ dados $x_i \in X$ e seus respectivos rótulos $y_i \in Y$, em que $X$ constitui o espaço dos dados e $Y = \{-1, +1\}$. $T$ é linearmente separável se é

possível separar os dados das classes +1 e -1 por um hiperplano (Figura 10). A equação de um hiperplano é apresentada na Equação 2.4, em que $w \cdot x$ é o produto escalar entre os vetores $w$ e $x$, onde $w \in X$ é o vetor normal ao hiperplano descrito e $\frac{b}{||w||}$ corresponde à distância do hiperplano em relação à origem, com $b \in \Re$.

$$f(x) = w \cdot x + b = 0 \tag{2.4}$$

Essa equação divide o espaço dos dados X em duas regiões: $w \cdot x + b > 0$ e $w \cdot x + b < 0$. Uma função sinal $g(x) = sgn(f(x))$ pode então ser empregada na obtenção das classificações, conforme ilustrado na Equação 2.5.

$$g(x) = sgn(f(x)) = \begin{cases} +1, & \text{se } w \cdot x + b > 0 \\ -1, & \text{se } w \cdot x + b < 0 \end{cases} \tag{2.5}$$

Os *SVMs de Margens Suaves* introduzem variáveis de folga $\xi_i$, para todo $i$ = 1,..., $n$, onde essas variáveis relaxam as restrições impostas ao problema de otimização primal:

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i, \;\; \xi_i \geq 0, \;\; \forall_i = 1, ..., n \tag{2.6}$$

A aplicação desse procedimento suaviza as margens do classificador linear, permitindo que alguns dados permaneçam entre os hiperplanos H1 e H2 e também a ocorrência de alguns erros de classificação. Um erro no conjunto de treinamento é indicado por um valor de $\xi_i$ maior que 1. Logo, a soma dos $\xi_i$ representa um limite no número de erros de treinamento. Para levar em consideração esse termo, minimizando assim o erro sobre os dados de treinamento, a Equação fica:

$$Minimizar_{w,b,\xi} \;\; \frac{1}{2} \; ||w||^2 \; + \; C \left( \sum_{i=1}^{n} \xi_i \right) \tag{2.7}$$

A constante C é um termo de regularização que impõe um peso à minimização dos erros no conjunto de treinamento em relação à minimização da complexidade do modelo. A presença do termo $\sum_{i=1}^{n} \xi_i$ no problema de otimização também pode ser vista como uma minimização de erros marginais, pois um valor de $\xi_i \in (0, 1)$ indica um dado entre as margens.

O problema de otimização gerado é quadrático, com as restrições lineares apresentadas na Equação 2.6. A sua solução envolve passos matemáticos com a introdução de uma função

Lagrangiana ($\alpha$) e tornando suas derivadas parciais nulas. Tem-se como resultado o seguinte problema dual:

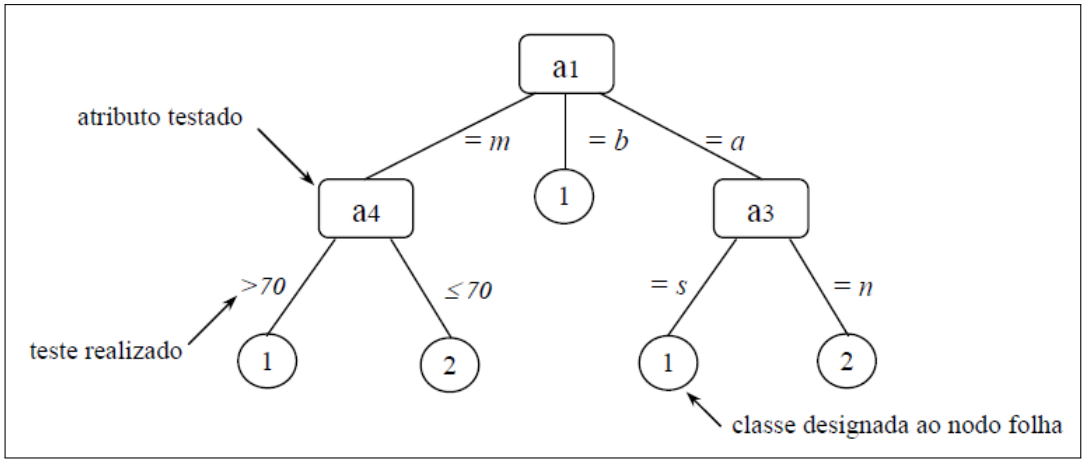
$$Maximizar_{\alpha} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \, (x_i \cdot x_j) \tag{2.8}$$

$$\text{Com as restrições: } \begin{cases} 0 \leq \alpha_i \leq C, \quad \forall_i = 1, ..., n \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases} \tag{2.9}$$

A escolha dos parâmetros do *kernel* do SVM são críticas para a obtenção de bons resultados. No caso específico de seleção dos parâmetros do *kernel* RBF, a busca é realizada em um espaço formado pelo parâmetro $\gamma$ e pelo parâmetro de custo $C$, onde o objetivo é encontrar uma melhor configuração para ambos, que leve à um bom desempenho do classificador. Esse trabalho utilizou $\gamma = 0.01$ e $C = 1.0$. Esses hiper-parâmetros podem ser calibrados utilizando ferramentas tais como o *grid-search* do *scikit-learn*[6].

***DT* (*Decision Tree*)**: Árvore de decisão é um algoritmo de classificação (ou regressão) constituído essencialmente de uma série de decisões *if-else*. Os dados são particionados em subconjuntos e alguma medida de pureza dos subconjuntos é avaliada para decidir quando parar. Um tratamento detalhado de DT para classificação pode ser encontrado em (ARTIFICIAL INTELLIGENCE IN MOTION, 2009).

**Figura 12 – Visualização de um classificador utilizando árvore de decisão**




Fonte: GARCIA (2003).

6 *Grid-search* do *scikit-learn* - Disponível em https://scikit-learn.org/stable/modules/grid_search.html

A Figura 12 mostra um classificador baseado em árvore de decisão, utilizando atributos quantitativos e categóricos. Os nodos são representados pelos atributos *a1, a3* e *a4*. Nos arcos são testados os valores do atributo designado ao nodo a quem pertence. Os testes são realizados de acordo com os valores dos atributos que, quando categóricos, são representados por uma igualdade, por exemplo, = *m*, onde *m* é um valor do atributo. Quando os valores são quantitativos, são representados por um intervalo de valor. Cada círculo ao final dos ramos da árvore, indica a classe associada aos nodos folha.

A árvore de decisão é construída por meio da aplicação de algoritmos de aprendizados sobre um conjunto de treinamento, sendo os mais conhecidos os algoritmos ID3, C4.5 e CART. Este último foi utilizado em (KALMEGH, 2015) para notícias e será utilizado nesse trabalho.

Algoritmo CART (*Classification and Regression Trees*): sua principal característica está na capacidade de gerar árvores com dimensões reduzidas, de alto desempenho, e com grande capacidade de generalização. Com a metodologia utilizada pelo CART na construção de árvores de decisão é concebível trabalhar com atributos previsores quantitativos ou categóricos. No particionamento utilizando atributos quantitativos, a técnica adotada é a pesquisa exaustiva do ponto de divisão; já quando utilizando atributos categóricos, são testadas todas as possibilidades de formação de dois subconjuntos com os possíveis valores.

A árvore resultante é baseada na técnica recursiva de divisão binária. É binário porque cada nodo é separado sempre em exatamente dois subconjuntos e a medida que se percorre a árvore, da raiz às folhas são respondidas questões simples do tipo sim/não. A recursividade se dá a cada subconjunto gerado, até que não seja mais possível ou necessário efetuar partições na árvore. A escolha da melhor característica pode ser feita com base em dois critérios: Gini (usado no trabalho) ou Entropia , e a atribuição de uma classe a cada folha é realizada com base no critério da classe mais provável ou da minimização dos custos. Um outro parâmetro utilizado no trabalho, foi o número mínimo de amostras, no caso 40, necessárias para dividir um nó interno.

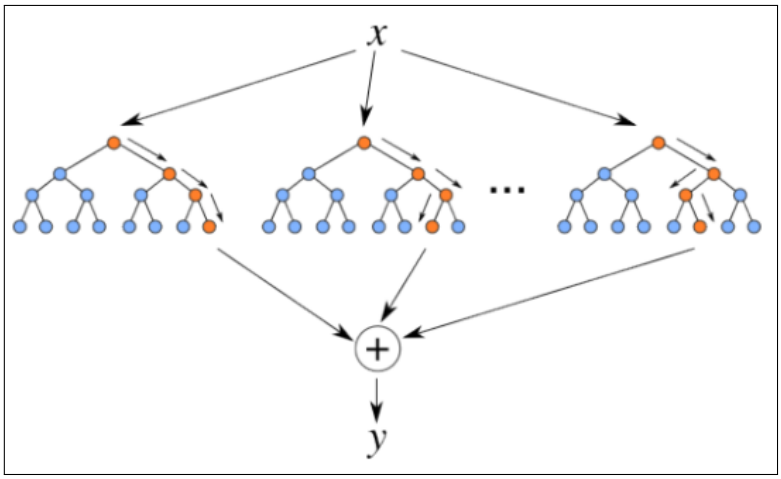
**RF (*Random Forest*)**: A Floresta Aleatória é um algoritmo classificador que faz uso do método de árvores de decisão possibilitando a mineração dos dados passados a mesma. Esta técnica possui uma ideia um pouco diferente dos algoritmos de árvores de decisão, a qual pertence. Enquanto uma árvore possui o objetivo de construção total de uma estrutura a partir de uma base de dados o *Random Forest* tem o objetivo de efetuar a criação de várias árvores de decisão usando um subconjunto de atributos selecionados aleatoriamente a partir do conjunto original. Estes subconjuntos são gerados por um tipo de amostragem chamado de *bootstrap*

(TELOKEN *et al.*, 2016), um tipo de amostragem com reposição, possibilitando assim melhor análise dos dados.

Com a quebra das massas de dados e construção de vários subconjuntos, uma árvore de decisão é construída. Com este procedimento, a construção das árvores ocorre pela seleção de atributos aleatoriamente a partir dos subconjuntos, onde os mesmos são aplicados nos nós de cada uma das árvores criadas. Uma RF é então um conjunto dessas árvores de decisão.

Após a criação do conjunto de árvores, é possível efetuar a classificação de qual delas possui melhor ganho de conhecimento para a solução de determinado problema. Para isto é necessário escolher o subconjunto de árvores de decisão que possui a melhor lógica e vantagens para a tomada de decisão. Para cada subconjunto é dado um voto sobre a qual classe o atributo chave deve pertencer, e este voto possui um "peso" o qual é afetado pela igualdade entre as árvores, "sendo que quanto menor a similaridade entre duas árvores melhor, e pela força que cada árvore tem individualmente, ou seja, quanto mais precisa uma árvore for, melhor será sua nota." (TELOKEN *et al.*, 2016).

**Figura 13 – Ilustração da lógica por trás do algoritmo *Random Forest***




Fonte: TELOKEN (2016).

As *Random Forests*, conforme verificado anteriormente, possuem a característica de Dividir-para–Conquistar, e isto possibilita a mesma algumas características que se destacam referentes às outras técnicas. Segundo (TELOKEN *et al.*, 2016) algumas dessas são :

- Algoritmo mais poderoso do que comparado somente a uma árvore de decisão;
- Possui boa taxa de acerto quando testado em diferentes conjuntos de dados;
- Evitam sobre ajuste (*overfitting*);
- Classificação aleatória das árvores sem intervenção humana.

Na Figura 13 é apresentado o funcionamento do *Random Forest*. É possível verificar que partindo de um elemento X, em uma base de dados, gerou-se várias *Random Forest*. Neste ponto cada uma gera várias regras e nelas a possibilidade de descoberta de novos padrões que poderão ser decisivos na tomada de decisão correta. Com as florestas criadas o próximo passo é calcular qual delas contém as regras mais exatas para a mineração. Com a escolha feita, ela é aplicado na base de dados e assim chegando a um resultado Y.

### 2.2.3 Avaliação em Classificação

A avaliação é uma etapa importante para o desenvolvimento de um método de classificação de texto. Sem a devida avaliação e uma análise comparativa, não há como determinar o quão bom é um classificador. Nesta seção, descreve-se as métricas utilizadas para avaliar as soluções empregadas nesse trabalho.

O resultado mais básico de um processo de classificação é representado pela Matriz Confusão que mostra o desempenho de um algoritmo de classificação a partir da relação de qual(is) registro(s) do conjunto de dados foram classificados corretamente - é a base para os cálculos das métricas dela derivadas. (PASCOAL *et al.*, 2016). A Tabela 1 mostra a forma de uma Matriz de Confusão para classificação binária.

**Tabela 1 – Matriz de Confusão para classificação binária**

| **Classes dos Dados** | **Classificado como Positivo** | **Classificado como Negativo** |
|---|---|---|
| **Positivo** | Verdadeiro Positivo (VP) | Falso Positivo (FP) |
| **Negativo** | Falso Negativo (FN) | Verdadeiro Negativo (VN) |

Fonte: Elaborado pelo autor.

Na Recuperação de Informação, *Precision (P)*, *Recall (R)*, *F1-Score (F1)* e Acurácia (AC) são métricas padrões adotadas para avaliar a qualidade de saída do classificador. Elas são definidas a seguir a partir de informações da Matriz de Confusão.

*Precision* é uma medida da relevância do resultado, ou seja, a fração dos resultados retornados que são relevantes.

$$Precision\ (P) = \frac{VP}{VP+FP} \tag{2.10}$$

*Recall* é uma medida de quantos resultados realmente relevantes são retornados.

$$Recall\ (R) = \frac{VP}{VP+FN} \tag{2.11}$$

A *F1-Score (F1)* é interpretada como uma média ponderada das métricas *precision e recall*.

$$F1\text{-}Score\ (F1) = \frac{2 * P * R}{P + R} \tag{2.12}$$

A Matriz de Confusão da Tabela 1 e cada uma dessas definições pode ser estendida para classificação com múltiplas classes bastando para isso considerar uma das classes na tabela como a classe de referência e a outra representando o conjunto das outras classes.

Métricas convencionais, para classificação binária, como *Precision (P)*, *Recall (R)* e *F1-Score (F1)* descritas acima, precisam ser adaptadas para o cenário de classificação monorrótulo e multirrótulo. Duas maneiras são comumente utilizadas para fazer essa adaptação. Na **macro-média** utiliza-se calcular as medidas individualmente para cada rótulo e fazer a média dessas medidas. Na **micro-média**, por outro lado, o procedimento é somar os valores para cada rótulo ($t$) e calcular o valor das medidas a partir dessa somatória.

Especificamente, *Precision (P)*, *Recall (R)* e *F1-Score (F1)* para classificação monorrótulo e multirrótulo, usando **micro-média** ($\mu$), são utilizados neste trabalho e podem ser descritas como em (SOKOLOVA; LAPALME, 2009):

$$Precision\ (P_\mu) = \frac{\sum_{i=1}^{t} VP_i}{\sum_{i=1}^{t} (VP_i + FP_i)} \tag{2.13}$$

$$Recall\ (R_\mu) = \frac{\sum_{i=1}^{t} VP_i}{\sum_{i=1}^{t} (VP_i + FN_i)} \tag{2.14}$$

$$F1\text{-}Score\ (F1_\mu) = \frac{2 * P_\mu * R_\mu}{P_\mu + R_\mu} \tag{2.15}$$

Uma outra métrica frequentemente utilizada na recuperação da informação é a acurácia que é a fração das decisões (positivo/negativo) que são corretas.

$$AC = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.16}$$

**Validação Cruzada** - Em todas as avaliações de classificadores realizadas neste trabalho foi utilizado o seguinte procedimento: os dados disponíveis foram separados em conjunto de projeto (*design*) e conjunto de validação. No conjunto de projeto foi utilizada validação cruzada para ajuste de parâmetros dos algoritmos, chamado de seleção de modelo. O desempenho do modelo assim selecionado foi então avaliado utilizado o conjunto de validação.

A validação cruzada (*cross validation*) se destina principalmente a ser utilizada na fase de projeto de classificadores. Entretanto, quando o custo computacional permite, ela tornou-se também um método padrão para garantir a validação estatística dos resultados da classificação bastando para isso que os dados sejam embaralhados em um certo número de corridas.

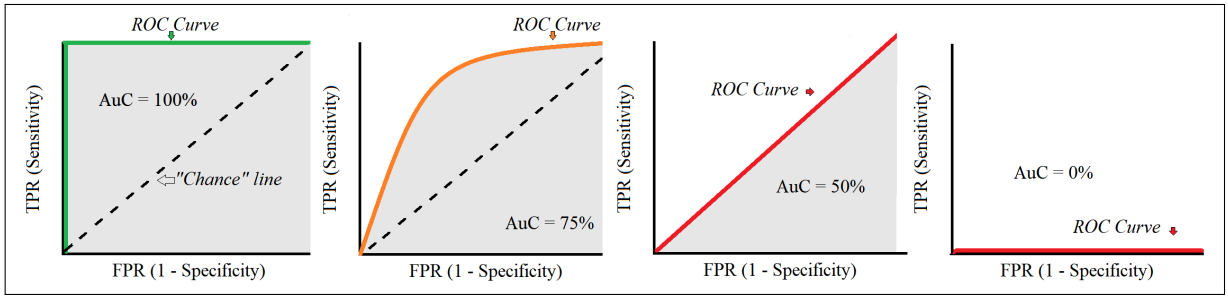
Ela consiste em construir $k$ diferentes classificadores: $\Psi_1, \Psi_2, ..., \Psi_k \cdot$ Para validar o classificador, dividimos o conjunto de documentos de treinamento $D_t$ em $k$ conjuntos disjuntos ou *folds** de tamanhos: $N_{t1}, N_{t2}, ..., N_{tk}$. O classificador $\Psi_i$ usa o *i-ésimo fold*, de tamanho $N_{ti}$, como conjunto de teste, e os documentos restantes em $D_t$, de tamanho $N_t$ - $N_{ti}$, como conjunto de treinamento (BAEZA-YATES; RIBEIRO-NETO, 2013).

Cada classificador é avaliado de maneira independente, usando alguma métrica de avaliação, tal como valores de *f1-score*, no caso do trabalho. A validação cruzada é feita comumente pela computação da média das $k$ medidas. No trabalho além da média foi calculado também a variância. O valor mais utilizado, na literatura, de $k$ é 10 (usado no trabalho), e, em tal situação, o método é chamado de validação cruzada do tipo *10-fold*.

**Curvas ROC e AUC** - Estão entre as métricas mais aplicadas na avaliação de modelos em *Machine Learning* e serão descritas a seguir (SOUZA *et al.*, 2017).

A Curva ROC ("*Receiver Operating Characteristic*") é uma curva de compromisso. Ela mostra o quão bem o modelo criado pode distinguir entre duas alternativas (já que é utilizado para classificação). Essas duas alternativas pode ser 0 ou 1, ou positivo e negativo. Os melhores modelos podem distinguir com precisão entre as duas coisas que está a avaliar da maneira mais independente possível. Na Figura 14, vemos exemplos de comportamento de uma Curva ROC, onde a curva ideal na imagem da mais a esquerda preenche 100%, o que significa que você será capaz de distinguir entre resultados negativos e resultados positivos em 100% do tempo (o que é quase impossível na vida real). Quanto mais você for para a direita, pior será a detecção. A curva ROC para a extrema direita faz um trabalho pior do que o acaso, misturando os negativos e positivos (o que significa que você provavelmente tem um erro em sua configuração).

**Figura 14 – Demonstração de comportamento - Curva ROC**




Fonte: Adaptado do site Data Science Central (2019).

A ideia por trás da Curva ROC é que seja frequentemente possível melhorar a seletividade do classificador sacrificando a sensibilidade e vice versa. Assim, uma Curva ROC possui dois parâmetros (eixos):

- Taxa de verdadeiro positivo (*True Positive Rate - TPR*), que é dado por VP / (VP + FN);
- Taxa de falso positivo (*False Positive Rate - FPR*), que é dado por FP / (FP + VN).

Uma curva ROC traça "*True Positive Rate* vs. *False Positive Rate*" em diferentes limiares de classificação.

AUC ("*area under the ROC curve*") nada mais é que uma maneira de resumir a informação da curva ROC num único valor, agregando todos os limiares da ROC, calculando a "área sob a curva". O valor do AUC varia de 0.0 até 1.0. O limiar entre um classificador efetivo a classificação aleatória é 0.5. Ou seja, acima desse limite, o algoritmo contribui para a classificação correta. Quanto maior o AUC, melhor, como vemos na Figura 14.

Um modelo cujas previsões estão 100% erradas tem uma AUC de 0, enquanto um modelo cujas previsões são 100% corretas tem uma AUC de 1. Cada modelo apresentará um valor de AUC, o que irá auxiliar na escolha do melhor modelo.

Uma propriedade interessante do AUC é que a métrica é invariante em escala, uma vez que trabalha com precisão das classificações, em vez de seus valores absolutos. Além disso, ela também mede a qualidade das previsões do modelo, independentemente do limiar de classificação.

Um outro gráfico intimamente relacionado a Curva ROC é a **Curva *precision-recall***, esse gráfico resume o *trade-off* entre o TPR e o valor preditivo positivo para um modelo preditivo usando diferentes limites de probabilidade.

As curvas ROC são apropriadas quando as observações são balanceadas entre cada classe, enquanto a Curva *precision-recall* é apropriada para conjuntos de dados desequilibrados. Nos dois casos, a AUC pode ser usada como um resumo do desempenho do modelo.

**Raiz do Erro quadrático médio (RMSE)** (SANTANA, 2019) - RMSE é uma regra de pontuação quadrática que também mede a magnitude média do erro. Embora ela faça mais sentido para problemas de regressão ela pode também ser calculada em problemas de classificação uma vez que muitos classificadores geral saídas contínuas (escores) e a classificação é decidida por comparação a limiares. Ele é calculado pela Equação 2.17 e é a raiz quadrada da média das diferenças quadradas entre previsão e observação real (KIM; KIM, 2018).

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{2.17}$$

Para avaliação dos classificadores, foram utilizadas coleções de documentos próprias e coleções de referência, nas quais as classes corretas são conhecidas - o conjunto de teste (validação). Assim como no conjunto de treinamento (projeto), a atribuição de classes aos documentos na coleção de teste é feita por especialistas humanos. Além disso, os conjuntos de teste e de treinamento devem ser disjuntos para garantir que a avaliação seja baseada em documentos novos e não vistos. O conjunto de teste é usado para avaliar o classificador em um processo de duas etapas: (a) usar o classificador para atribuir classes aos documentos no conjunto de teste e (b) comparar as classes atribuídas pelo classificador com aquelas especificadas pelos especialistas humanos.

## 2.3 SISTEMAS DE RECOMENDAÇÃO

Os Sistemas de Recomendação (SR) são sistemas que geram recomendações individualizadas como saída, tendo como principal objetivo, guiar os usuários de maneira otimizada, orientada e customizada a objetos que lhe sejam úteis, diante de uma imensa variedade de opções (BURKE, 2007). Os Sistemas de Recomendação podem ser utilizados para recomendar conteúdo nos mais variados domínios, como por exemplo filmes, músicas, livros, artigos, viagens, varejo, notícias etc. Embora existam particularidades específicas em cada um dos domínios, dois papéis são imprescindíveis: os usuários do sistema e os conteúdos oferecidos pelo sistema. À vista disso, a base do SR é que o conteúdo oferecido deve ser filtrado com o objetivo de gerar uma lista customizada e relevante para cada um dos usuários.

Os recomendadores, independentemente das técnicas abordadas, baseiam-se no princípio da personalização da recuperação de informação de modo a agregar valor na experiência

do usuário. Este princípio da personalização tem forte dependência em como ele modela e identifica as preferências do usuário, pois é deste modelo que se infere as recomendações.

Os algoritmos de recomendação podem ser elaborados por meio de diversas abordagens para Sistemas de Recomendação, onde a decisão de qual abordagem utilizar é uma decisão de projeto e pode ser influenciada pelos mais variados fatores, tais como: Tipos de dados sobre os itens que estejam disponíveis, informações sobre os usuários e seus interesses, volume de usuários e itens, além da esparsidade da relação *usuário-item*, esses devem ser analisados minuciosamente para decidir qual tipo de abordagem empregar na solução de recomendação adotada no sistema. Algumas das abordagens mais utilizadas em sistemas de recomendação são as seguintes (SILVA, 2014):

**Baseada em Conteúdo (BC)** - Os recomendadores baseados em conteúdo, utilizam como princípio para realizar sua tarefa: analisar o conteúdo dos itens que foram avaliados pelo usuário e os demais itens do sistema para realizar as recomendações (PAZZANI, 1999). Os métodos de recomendação Baseados em Conteúdo têm origem nas áreas de RI (Recuperação de Informação) e AM (Aprendizado de Máquina, ou do inglês - *Machine Learning* (ML)). Resumidamente, as etapas executadas por um recomendador que baseia sua tarefa no cálculo de similaridade entre os atributos dos itens do sistema são descritas a seguir.

Nas recomendações por meio de conteúdo é necessário, primeiramente, construir uma representação para o conteúdo. Os dados podem ser estruturados, como data da notícia, categoria do produto, classificação ou gênero do filme e etc. Entretanto, é comum a presença do formato semiestruturado dos dados textuais (sinopse, descrição, título...) o que implica em estabelecer uma nova forma de representar os itens. Este problema é amplamente estudado em RI (SCHÜTZE; MANNING; RAGHAVAN, 2008), mas é fundamental na construção de um recomendador Baseado em Conteúdo. As representações de documentos mais utilizadas em sistemas de recomendações são feitas por meio de termos/palavras. Como visto anteriormente nesse capítulo, essas técnicas são conhecidas como *Bag-of-words (BoW)*, Frequência do termo x Inverso da Frequência do Documento (TF-IDF)) e técnicas utilizando *Word Embeddings* cada vez mais vem sendo aplicadas para essa abordagem, como em (NANDI *et al.*, 2018) e (SAMARINAS; ZAFEIRIOU, 2019).

Algumas das desvantagens da abordagem baseada em conteúdo são:

- Não é capaz de fornecer itens que não estejam de acordo com o conteúdo dos itens de interesse do usuário; (SHARDANAND; MAES, 1995);

- Termos/tópicos nem sempre podem ser utilizados para vincular um texto ao domínio de outro texto (DAS *et al.*, 2007). Há questões de forma de escrita ou de estilo autoral que podem fazer com que dois textos do mesmo domínio sejam classificados de forma diferente, como por exemplo: Brasil = País do Futebol.

**Baseada em Filtragem Colaborativa (FC)** - Diferentemente das técnicas que se baseiam em similaridade entre itens para recomendação, a Filtragem Colaborativa utiliza outro caminho para realização das recomendações. Seu princípio básico é identificar usuários semelhantes, ou seja, que tenham avaliado uma quantidade relevante de itens de modo semelhante e desta forma apresentar uns aos outros os itens distintos bem avaliados por cada um. Técnica essa nomeada como "automatização do boca-a-boca" por (SHARDANAND; MAES, 1995) e utilizada por (HU, 2019) , tem como objetivo uma filtragem na qual o trabalho de identificação do conteúdo relevante pode ser realizado de modo particionado dentre os usuários do sistema quando cada um aponta sua avaliação a um item recuperado.

Algumas das desvantagens da abordagem baseada em Filtragem Colaborativa, catalogadas no trabalho de (SU; KHOSHGOFTAAR, 2009), são:

- Esparsidade da relação Usuário X Item que quando presente pode dificultar a identificação dos vizinhos ou mesmo a qualidade dessa "vizinhança".
- Problema da partida a frio (*cold-start problem*), é composto por duas categorias, a primeira, o problema de novo usuário (*new user problem*) e consiste em como recomendar para usuários que possuem poucas ou nenhuma avaliação sobre os itens. A segunda categoria, o problema de novo item (*new item problem*), é referente a um novo item disponível no sistema que não possua avaliações dos usuários.

**Baseada em Contexto (CT)** - No decorrer dos últimos anos diversos trabalhos surgiram com a intenção de aprimorar os Sistemas de Recomendação, seja utilizando novos algoritmos ou adicionando novos elementos ao recomendador. Muitas pesquisas, como (ADOMAVICIUS *et al.*, 2011) e (ANAND; MOBASHER, 2006), apontam uma outra abordagem denominada "Baseada em Contexto". Nestes trabalhos, além de dados sobre os itens ou sobre os usuários, são utilizadas outras informações que agregam valor ao processo de filtragem. Em uma loja virtual que comercializa variedades, por exemplo, é importante identificar no perfil do usuário preferências de curto prazo (um pai que compra um artigo para seu filho recém nascido) e as preferências de longo prazo (para um atleta, oferecer artigos esportivos). Além disso,

informações sobre localização geográfica, orientação religiosa ou política dos usuários podem ajudar o recomendador a incrementar a qualidade de suas recomendações. Estas informações extra são chamadas informações de contexto.

**Híbrida (HB)** - Há um aspecto complementar entre as principais abordagens de recomendação, pois justamente onde uma técnica limita-se a outra avança, transformando este fato em uma oportunidade. Em (KHATTAR *et al.*, 2018) é apresentada uma aplicação que visa a utilização de duas abordagens de modo à complementá-las. Este tipo de sistema é denominado híbrido e pode, por exemplo, eliminar o problema de partida a frio da Filtragem Colaborativa recomendando um item similar a outro visualizado (baseado em conteúdo), assim como surpreender o usuário apresentando itens de interesse de outro usuário com gosto similar. Em (BURKE, 2007) é apresenta uma taxonomia dos Sistemas de Recomendações Híbridos, estes são comparados e apresentado alguns cenários de aplicação para os mesmos.

### 2.3.1 Sistema de Recomendação de Notícias

A recomendação de notícias em geral representa outro domínio de aplicação no qual várias das técnicas conhecidas para criar recomendações automatizadas podem ser aplicadas. De fato, tem havido um número de casos relatados em que tais sistemas estão sendo usados em ambientes "online" para fornecer recomendações de notícias em sites populares (LIU; DOLAN; PEDERSEN, 2010) e (LOMMATZSCH; ALBAYRAK, 2015). No entanto, os problemas de recomendação de notícias geralmente apresentam certas características que não estão presentes ou pelo menos não são relevantes em outros domínios. Por exemplo, em contraste com outros domínios como a recomendação de filmes, a relevância dos itens de notícias pode mudar muito rapidamente, ou seja, diminuir ou aumentar novamente devido a eventos reais recentes (AHN *et al.*, 2007), (ÖZGÖBEK; GULLA; ERDUR, 2014) e a “rotatividade de itens” é comumente alta (DAS *et al.*, 2007). Na verdade, como os sites de notícias são frequentemente atualizados, alguns artigos podem ser substituídos por um artigo de “notícias de última hora” sobre o mesmo tópico várias vezes durante um mesmo dia, o que pode exigir constantes atualizações nos modelos de recomendação.

Alguns dos desafios típicos no domínio de notícias são que o interesse de um usuário pode mudar dinamicamente, dependendo de diferentes fatores contextuais, como a hora do dia, os recursos do dispositivo do usuário (por exemplo: celular, tablet, computador) ou a localização

atual do usuário (GULLA *et al.*, 2013), (MA; LIU; SHEN, 2016). Usuários geralmente não estão dispostos a avaliar a relevância das notícias tendo o SRN que interpretar sinais implícitos para aprender os seus interesses e satisfação, como a abordagem baseada em conteúdo, por exemplo em (MUSTO *et al.*, 2016) e (NANDI *et al.*, 2018), além de lidar com dados não estruturados que podem ser difíceis de interpretar, incompletos e inconsistentes.

# 3 TRABALHOS RELACIONADOS

Neste capítulo serão revisadas as ideias de alguns trabalhos encontrados na literatura que tratam do uso de abordagens *embeddings* para sistemas de recomendação.

Muita pesquisa tem sido realizada nos últimos anos em sistemas de recomendação e recomendação de notícias personalizadas. Alguns sistemas usam métodos clássicos como filtragem colaborativa combinado com o método *Latent Dirichlet Allocation* (LDA) para aprender o conteúdo (GARCIN *et al.*, 2012). O Google Notícias usou a filtragem colaborativa combinada com uma estrutura Bayesiana que modela o comportamento de clique dos usuários (LIU; DOLAN; PEDERSEN, 2010). Outro sistema proposto usa filtragem baseada em conteúdo usando palavras-chave dos artigos e constrói um perfil para cada usuário com essas palavras-chave, e compara o método proposto com o TF-IDF (KOMPAN; BIELIKOVÁ, 2010). Em (OH *et al.*, 2014), o mesmo método é usado, mas as palavras-chave são filtradas usando uma rede neural. Uma representação de documento adaptada do TF-IDF é comparada com um modelo proposto baseado no TF-IDF para tarefa de classificação do Perfil do usuário. Para encontrar notícias recomendáveis para cada usuário com base no perfil do usuário utilizam *Bag-of-Words* (BoW), onde é calculado a similaridade cosseno do *bag-of-words* de cada notícia com os perfis de usuários.

O *framework* proposto em (ZHENG *et al.*, 2013) usa agrupamento hierárquico conjunto que separa os usuários em diferentes grupos com base em suas histórias de leitura, onde cada usuário pode pertencer a vários grupos. Para organizar o perfil do usuário, eles usam o *Latent Dirichlet Allocation*, um modelo que é usado com muita frequência para modelagem de tópicos (NANDI *et al.*, 2018). Às vezes, em vez de uma abordagem baseada em termos, uma abordagem baseada em ontologia é adotada, como em (IJNTEMA *et al.*, 2010).

Em trabalhos mais recentes, as redes neurais recorrentes são usadas para recomendação de notícias baseada em sessões (HIDASI *et al.*, 2015). A sequência de artigos visualizados é fornecida como entrada (a sessão do usuário) e os IDs dos artigos são fornecidos para a saída (as recomendações). Uma variação dessa abordagem adiciona uma camada recorrente de GRU (*Gated Recurrent Units*) adicional para modelar informações em sessões de usuários e acompanhar a evolução dos interesses do usuário ao longo do tempo (QUADRANA *et al.*, 2017).

Outra técnica que vem ganhando cada vez mais atenção devido ao seu bom desempenho para sistemas de recomendação são as Representações de Documentos baseadas em "casamento de Palavras" *(Word Embeddings)*. Representações de Documentos são utilizadas

para modelar a língua e possibilitar que a máquina a entenda. Para obtê-las são necessários pré-processamentos que abstraem e estruturam a língua, deixando apenas o que é informação relevante. Como já dito antes, essas informações são mapeadas normalmente em um vetor de característica. Na seção 2.1.2 foi exposto mais detalhes sobre Representações de Documentos.

Os trabalhos recentes tendem para a adoção de técnicas baseadas em incorporação de palavras (*word embeddings*) em cenários de recomendação. Em (MUSTO *et al.*, 2016), a abordagem para recomendação empregada foi a baseada em conteúdo, onde utilizam o *Word2Vec* para criar um perfil de usuário. Os resultados experimentais mostram que um modelo baseado em *word embeddings* é comparável ao de algoritmos de bom desempenho baseados em Filtragem Colaborativa e Fatoração de Matriz.

Em (KARATZOGLOU; HIDASI, 2017) os autores encorajam o uso de combinação dessas técnicas. E é através da combinação de duas técnicas que o ***Luppar News-Rec*** é construído.

**RECOMENDAÇÃO DE NOTÍCIAS** - No que se diz respeito a recomendação de notícias, em (SAMARINAS; ZAFEIRIOU, 2019) é apresentado a arquitetura de um sistema de recomendação de notícias, baseado em conteúdo (não-supervisionado) utilizando o modelo *Word Embeddings - FastText* para fornecer resultados para os usuários. Eles introduziram um método para modelar os interesses dos usuários ao longo do tempo e uma estrutura para filtrar e marcar notícias de alta qualidade usando modelos de classificação de texto baseados em uma arquitetura de rede neural Bi-LSTM (*Bidirectional-Long Short Term Memory*) e agrupamento de notícias (*Agglomerative clustering*).

As abordagens tradicionais de Filtragem Colaborativa (*Collaborative Filtering* - CF) exploram apenas as relações de itens do usuário (por exemplo, cliques, curtidas e visualizações), portanto, elas sofrem com o problema da dispersão de dados. Os itens geralmente são associados a textos não estruturados, como resumos de artigos e análises de produtos. Em (HU, 2019) desenvolve-se uma estrutura personalizada de Neural *Embedding* (PNE) para explorar as interações e as palavras de forma integrada. O PNE consiste em uma rede CF (CFNet) para aprender uma função de interação não-linear e de uma rede de memória (MemNet) para combinar a semântica da palavra com as preferências do usuário. O objetivo foi aprender essas incorporações de usuários, itens e palavras em conjunto e prever as preferências do usuário em itens com base nessas representações aprendidas. O PNE estima a probabilidade de um usuário gostar de um item por dois fatores - fatores de comportamento e fatores semânticos. Em dois

conjuntos de dados do mundo real (*Public Amazon Products* e *Cheetah Mobile News*), o PNE desenvolvido mostra um desempenho melhor do que quatro *baselines* de última geração em termos de três métricas de classificação. Também mostram que o PNE aprende a incorporação de palavras (*word embeddings*) significativas por visualização.

Uma abordagem híbrida é apresentada em (KHATTAR *et al.*, 2018), baseada em redes neurais para recomendação de notícias. Essa abordagem incorpora tanto a interação de item do usuário (Filtragem Colaborativa) quanto a informação de conteúdo dos artigos lidos pelo usuário no passado (baseada em conteúdo). Primeiramente criaram um perfil baseado em artigo para o usuário. Em seguida, usaram esse perfil de usuário com amostras positivas e negativas adequadas para treinar o modelo baseado na rede neural. Utilizam como *Embedding* o *doc2vec* com vetor de 300 dimensões. O modelo resultante é então aplicado em um conjunto de dados do mundo real (*CLEF NewsReel 2017*). Compararam com um conjunto de *baselines* de base estabelecidas e os resultados experimentais mostram que o modelo supera o estado da arte.

Já o trabalho de (MOREIRA; FERREIRA; CUNHA, 2018) apresenta uma arquitetura *Deep Learning* para recomendações baseadas em sessão de artigos de notícias, o *DeepRecNews*. Esta arquitetura é composta por dois módulos, sendo o primeiro responsável por aprender representações de artigos de notícias, com base em seu texto e metadados, e o segundo módulo teve como objetivo fornecer recomendações baseadas em sessão usando Redes Neurais Recorrentes, utilizando *embeddings* pré-treinados (*Word2Vec, Glove*) em português. A tarefa de recomendação abordada é a previsão do próximo item para sessões do usuário, utilizando rede neural LTSM (*Long Short-Term Memory*). O contexto da sessão do usuário é aproveitado pela arquitetura para fornecer informações adicionais em um cenário de inicialização a frio. O comportamento dos usuários e os recursos dos itens são mesclados em uma abordagem de recomendação híbrida.

Finalmente, em (NANDI *et al.*, 2018) é apresentado um sistema recomendador de notícias utilizando a abordagem baseada em conteúdo, aplicado ao *Bangla*[1] usando vetores de parágrafo também conhecidos como *doc2vec*, uma extensão do *Word2Vec*. O *doc2vec* é uma abordagem conduzida pela rede neural que encapsula a representação do documento em um vetor de baixa dimensionalidade. O *doc2vec* pode capturar efetivamente o relacionamento semântico entre documentos de uma grande coleção de textos. Foram realizados experimentos qualitativos e quantitativos em um grande *corpus* de notícias *Bangla* e mostrado que o *doc2vec* tem um desempenho melhor do que duas técnicas populares de modelagem de tópicos como LDA (*Latent Dirichlet Allocation*) e LSA (*Latent Semantic Analysis*). No cenário das 10 melhores

[1] Site de Notícias de Bangladesh - Bangla (https://www.bn.bangla.report/)

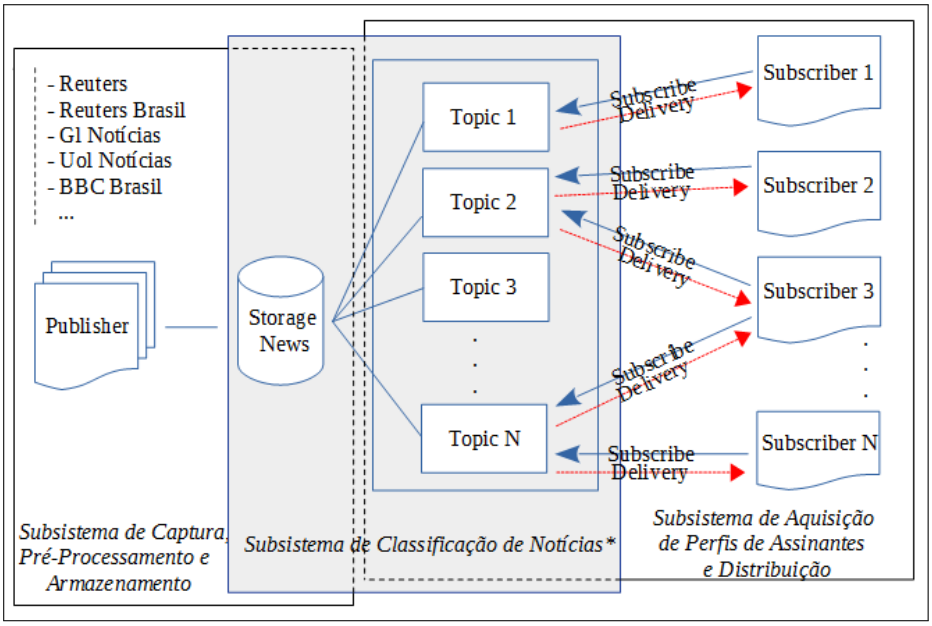
recomendações, as sugestões do *doc2vec* são mais contextualmente corretas do que o LDA e o LSA. O *doc2vec* também supera o LDA e o LSA em conjuntos de dados tripletos gerados por humanos com 91% de precisão, nos quais o LDA e o LSA fornecem 85%, 84% de precisão, respectivamente.

Pode-se notar por essa revisão de um grupo de trabalhos relacionados que muitas ideias de combinação de métodos já foram propostas e estudadas. Neste trabalho, para a tarefa de recomendação de notícias foi utilizada a abordagem baseada em conteúdo, utilizando subscrição (onde apenas leva em conta os tópicos escolhidos pelo usuário, e não baseado no que o usuário já acessou, como é o caso do (NANDI *et al.*, 2018)). Além da representação *embedding - Word2Vec*, também foi utilizado a representação *FastText*, como em (SAMARINAS; ZAFEIRIOU, 2019). Ambas as abordagens *embedding* utilizando vetor de 300 dimensões como em (KHATTAR *et al.*, 2018). O *embedding (Word2Vec)* pré-treinado em Português utilizado nesse trabalho, foi o mesmo utilizado em (MOREIRA; FERREIRA; CUNHA, 2018). Além do tradicional vetor média para representações *embeddings*, uma abordagem ponderada por IDF (*inverse document frequency*) foi estudada e seu desempenho avaliado.

# 4 PROJETO DO SISTEMA

Neste capítulo será apresentado a proposta de desenvolvimento do SRN baseado em *publish/subscribe*, o qual se chamará de ***Luppar News-Rec***[1], que servirá de suporte aos objetivos do trabalho. O ***Luppar News-Rec*** é um SRN composto por algoritmos clássicos de classificação que trabalham em conjunto com representações de documentos para solucionar o problema de classificação de notícias de forma a trazer os documentos (notícias) que atendam a necessidade do usuário. A metodologia que segue esse trabalho divide os esforços da implementação do SRN em três etapas: Subsistema de Captura, Pré-Processamento e Armazenamento, Subsistema de Classificação de Notícias e Subsistema de Aquisição de Perfil de Assinantes e Distribuição. Segue-se descrevendo cada etapa.

**Figura 15 – Estrutura do SRN baseado em *Publish-Subscribe*, apresentando os três subsistemas que o compõe**




Fonte: Elaborado pelo autor.

**O SRN BASEADO EM PUBLISH-SUBSCRIBE** - A estrutura do SRN é apresentada na Figura 15. A ideia central do sistema é utilizar o paradigma de comunicação *publish/subscribe*, onde os produtores (como por exemplo: *Reuters, Inshorts*, G1 Notícias, *Reuters* Brasil, UOL Notícias, Folha de São Paulo, *BBC News* Brasil etc) publicam as notícias (*publish*) enquanto os leitores/usuários, aqui chamados assinantes (*subscriber*), inscritos em tópicos do seu interesse passam a receber (*delivery*) notícias apenas desses tópicos.

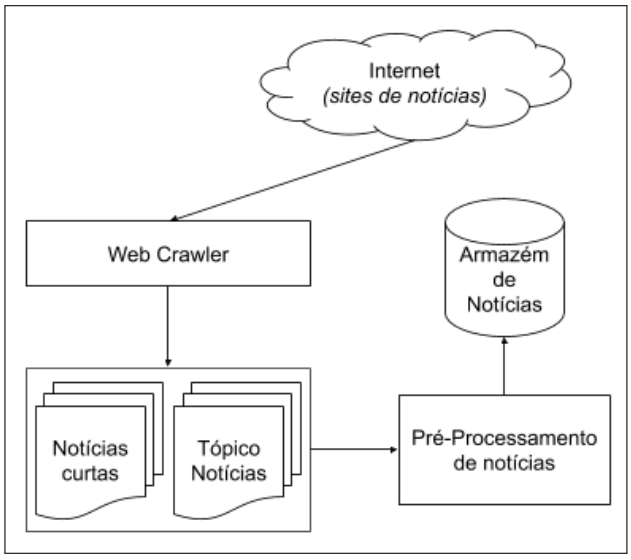
[1] Uma interface de teste do ***Luppar News-Rec*** pode ser acessado no endereço: http://luppar.com/recommender

O paradigma de comunicação *publish/subscribe* tem propriedades interessantes para esta aplicação, sendo uma delas o desacoplamento entre produtores e assinantes em termos de tempo, espaço e sincronização. Em *publish/subscribe* os geradores de notícias e os assinantes não necessitam estar disponíveis ao mesmo tempo. Isso significa que um assinante pode receber as notificações mesmo que um novo produtor seja criado para determinado contexto. Tal característica tem alavancado o uso de aplicações com comunicação *publish/subscribe*, principalmente no contexto de computação móvel, onde a existência de uma conexão muitas vezes não pode ser garantida. Além disso, os eventos publicados são disponibilizados para os assinantes interessados, sem os produtores necessariamente conhecerem os assinantes ou sua localização, e vice-versa (ADHIANTO *et al.*, 2010).

A Figura 15 mostra em blocos o relacionamento entre os três subsistemas: Captura, Pré-Processamento e Armazenamento de Notícias (*storage news*), classificação de notícias e aquisição de perfis de assinantes e distribuição.

## 4.1 SUBSISTEMA DE CAPTURA, PRÉ-PROCESSAMENTO E ARMAZENAMENTO

**Figura 16 – Processo automático de coleta e transformação de notícias (EN / PT-BR)**




Fonte: Elaborado pelo autor.

A Figura 16 exibe o funcionamento deste subsistema, que tem como objetivo a coleta de notícias, tanto no funcionamento *Online* quanto para a construção de coleções de documentos para aprendizagem supervisionada. Todo esse processo é automatizado através de scripts fazendo uso de rastreadores web (*Web Crawler*).

Nos experimentos, as notícias curtas (lide[2] de notícias) disponibilizadas em sites de notícias (G1, Reuters, BBC Brasil, Estado de São Paulo...) em um formato não estruturado, são baixadas e em seguida o Pré-Processamento é aplicado. Em (SANTOS; GATTI, 2014) um texto curto é definido como um texto até 10 palavras. Essa definição cobre um certo número de aplicações tais como mensagens do *Twitter*, a maioria dos comentários são muito curtos e de opiniões sobre produtos, além de lides (*lead*) de notícias. Entretanto, para efeito desta dissertação considerou-se texto como aqueles de até 70 palavras para englobar notícias curtas que vão além da lide, comentários e opiniões mais longas e resumos em geral de textos maiores.

Na fase de Pré-Processamento, são removidas pontuações e numerações, aplicado tokenização (marca cada palavra como um *token* no texto), remoção de *stopwords* (palavras com baixo poder discriminativo (artigos, preposições)) e aplicação de *stemming* (radicalização das palavras).

Um conjunto de notícias capturado e armazenado é tratado como uma coleção de documentos. O projeto admite gerenciar múltiplas coleções de documentos. Nos experimentos, a primeira etapa deste subsistema foi obter as coleções de documentos próprias. Foram criadas coleções de documentos contendo notícias curtas extraídas de sites de notícias em Inglês e em Português, devidamente distribuídas em tópicos (5, 12 e 20 tópicos - detalhados no APÊNDICE B), e são descritas a seguir:

- **Z5News** - Contém 42.532 notícias curtas em inglês, devidamente rotuladas e distribuídas em 5 tópicos distintos (*sportsnews, politicsNews, technologyNews, PersonalFinance and brazil-news*), extraídas do site de notícias: *Reuters* (entre os anos de 2019 e 2011);
- **Z5NewsBrasil** - Contém 22.177 notícias curtas em português, devidamente rotuladas e distribuídas em 5 tópicos distintos (Esporte, Política, Tecnologia, Finanças e Educação), extraídas do site de notícias: G1 notícias[3] (entre os anos de 2019 e 2011);
- **Z12News** - Contém 41.849 notícias curtas em inglês, devidamente rotuladas e distribuídas em 12 tópicos distintos (*sportsnews, politicsNews, technologyNews, PersonalFinance, brazil-news, aerospace-defense, autos, commoditiesNews, fundsNews, foreignexchange-News, healthnews and environmentnews*), extraídas do site de notícias: *Reuters* (entre os anos de 2019 e 2011);

[2] Em jornalismo, o lide (em inglês: *lead*) é a primeira parte de uma notícia.
[3] *G1 notícias* - https://g1.globo.com/

- **Z20News** - Contém 117.027 notícias curtas em inglês, devidamente rotuladas e distribuídas em 20 tópicos distintos (*sportsnews, politicsNews, technologyNews, PersonalFinance, brazil-news, businessNews, innovationNews, sustainable_business, aerospace-defense, autos, mcbreakingviews, banks, marketsNews, commoditiesNews, fundsNews, foreignexchangeNews, wealth-retirement, worldNews, healthnews and environmentnews*), extraídas do site de notícias: *Reuters* (entre os anos de 2019 e 2011).

Para os testes do SRN foram baixadas notícias em inglês dos sites: *Reuters*[4] e *Inshorts*[5], assim como notícias em português dos sites: *Reuters* Brasil, UOL Notícias, BBC Brasil, Estado de São Paulo dentre outros, através do agregador de notícias da UOL[6]. Além dessas, esse subsistema tem a função de diariamente coletar, pré-processar e armazenar notícias desses sites para servir de insumo para os próximos subsistemas que compõe o SRN.

A Tabela 2 mostra as características dessas coleções.

**Tabela 2 – Características das coleções de notícias próprias**

| Coleção | Idioma | Tipo | Notícias | Tópicos | Termos (média) | Finalidade |
|---|---|---|---|---|---|---|
| Z5News | EN | Monorrótulo | 42.532 | 5 | 22 | Treinamento |
| Z5News (*Offline*) | EN | Monorrótulo | 769 | 5 | 31 | Teste *Offline* |
| Z5News (*Online*) | EN | Monorrótulo | 187 | 5 | 27 | Teste *Online* |
| Z5NewsBrasil | PT | Monorrótulo | 24.177 | 5 | 15 | Treinamento |
| Z5NewsBrasil (*Offline*) | PT | Monorrótulo | 656 | 5 | 17 | Teste *Offline* |
| Z5NewsBrasil (*Online*) | PT | Monorrótulo | 189 | 5 | 18 | Teste *Online* |
| Z12News | EN | Monorrótulo | 41.849 | 12 | 25 | Treinamento |
| Z12News (*Offline*) | EN | Monorrótulo | 1.360 | 12 | 24 | Teste *Offline* |
| Z12News (*Online*) | EN | Monorrótulo | 232 | 12 | 28 | Teste *Online* |
| Z20News | EN | Monorrótulo | 117.027 | 20 | 22 | Treinamento |
| Z20News (*Offline*) | EN | Monorrótulo | 2.246 | 20 | 24 | Teste *Offline* |
| Z20News (*Online*) | EN | Monorrótulo | 376 | 20 | 29 | Teste *Online* |

Fonte: Elaborado pelo autor.

## 4.2 SUBSISTEMA DE CLASSIFICAÇÃO DE NOTÍCIAS (*CORE*)

Esse subsistema tem como entrada as notícias coletadas, pré-processadas e armazenadas na etapa anterior. O objetivo é classificar essas novas notícias em tópicos pré-estabelecidos.

[4] *Reuters* - https://www.reuters.com/news/archive
[5] *Inshorts* - https://inshorts.com/en/read
[6] *UOL Notícias* - https://noticias.uol.com.br

Os classificadores KNN - *K Nearest Neighbor*, SVM - *Support Vector Machines (Kernel: RBF)*, DT - *Decision Tree* e RF - *Random Forest* foram combinados com as representações de documentos BoW, TF-IDF, *Word2Vec* e *FastText*, além da abordagem `E2V-IDF`, aplicada ao *Word2Vec* e *FastText* para investigar as melhores soluções.

A primeira etapa desse subsistema, o *core* do ***Luppar News-Rec***, foi a realização de uma avaliação *Offline* de todas as combinações Classificador X Representação, e a seleção das três combinações mais bem avaliadas. Os resultados dessa etapa serão apresentados Seção 5.1.

Os classificadores e representações de documentos mais bem avaliados foram implementados neste subsistema do ***Luppar News-Rec*** e posteriormente avaliados em novas coleções de notícias incluindo notícias de outras fontes.

## 4.3 SUBSISTEMA DE AQUISIÇÃO DE PERFIS DE ASSINANTES E DISTRIBUIÇÃO

Embora previsto em projeto, a aquisição de perfis de usuários não foi implementado neste trabalho. Como explicado no Capítulo 2, técnicas de filtragem colaborativa, rastreamento de interesse por conteúdo, verificação de contexto ou um híbrido delas podem ser utilizadas para isso. Nesta implementação a aquisição do interesse dos assinantes é feita via informação direta do usuário. A partir de então a função do subsistema é apenas de validação, autenticação e distribuição.

Neste subsistema é implementado o Recomendador em si, utilizando a abordagem baseada em conteúdo (subscrição em tópicos) que tem como objetivo a distribuição dos documentos (notícias), atribuídos a tópicos na etapa de classificação, para os usuários que assinaram os determinados tópicos.

A subscrição em tópicos implementada na versão atual do ***Luppar News-Rec*** funciona da seguinte forma:

1. O usuário inscreve-se e informa o e-mail no qual deseja receber as notícias;
2. O ***Luppar News-Rec*** oferece um catálogo de tópicos e/ou subtópicos;
3. O usuário "assina" (*subscribe*) os tópicos de seu interesse;
4. O ***Luppar News-Rec*** diariamente entregará (*delivery*), via e-mail, aos assinantes as notícias dos tópicos assinados por esses.

O sistema armazena o perfil de cada um dos assinantes (e-mail e tópicos escolhidos - exemplo no APÊNDICE A), e esses dados serão utilizados para avanço do ***Luppar News-Rec*** para uma abordagem híbrida em futuros trabalhos, ou seja, partindo também para o uso de

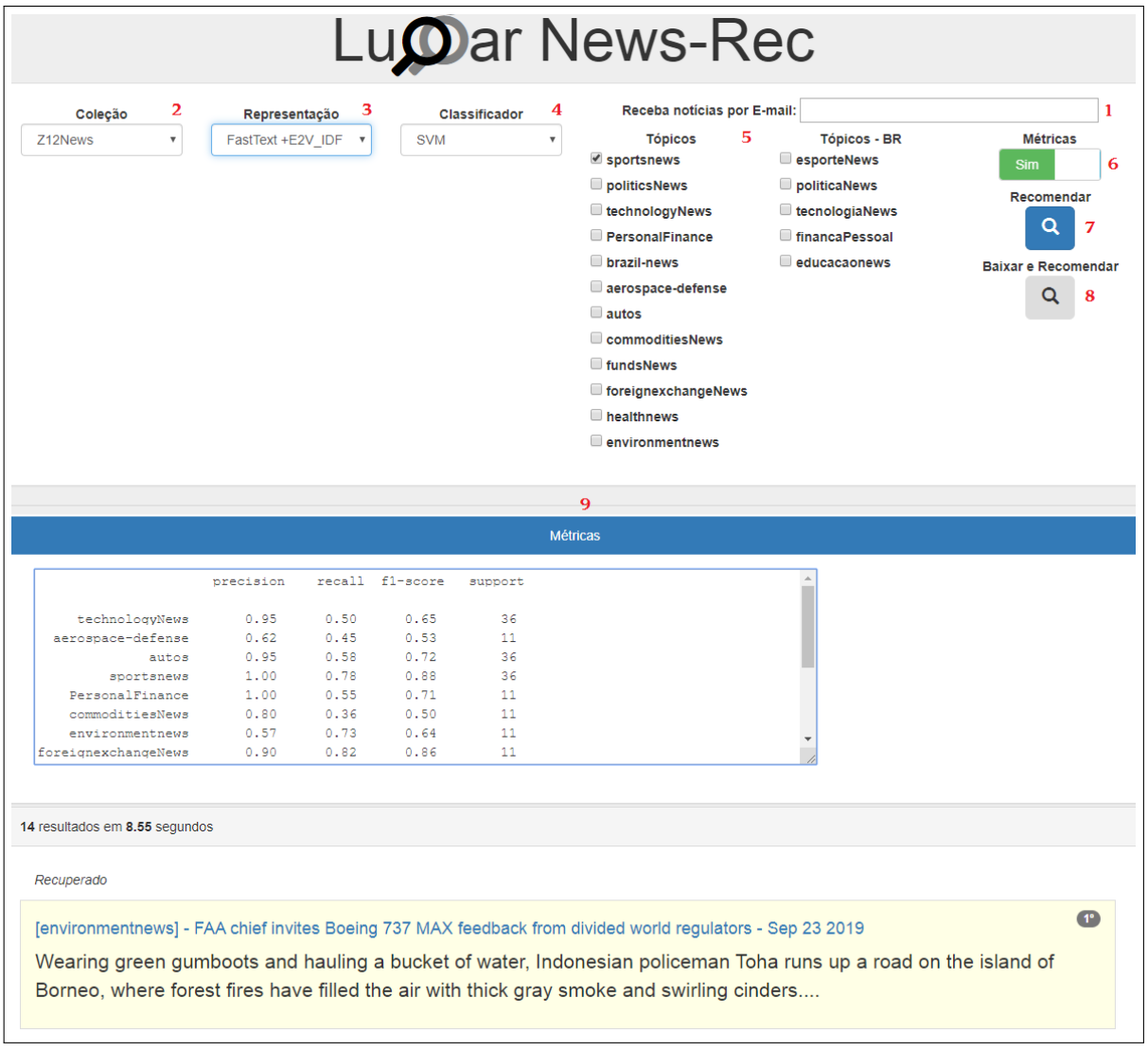
Filtragem Colaborativa (aprendizagem de perfil), indicando notícias com base em perfis de assinantes semelhantes.

### 4.3.1 Interface

Embora não tenha sido produzida ainda uma versão comercial do ***Luppar News-Rec***, foi construída uma interface que tem como principal objetivo a realização de testes e demonstrações de resultados, como apresentada na Figura 17.

Coleções com 24 horas de notícias curtas de algumas fontes são construídas pelo subsistema de aquisição e armazenadas. Essa interface permite a configuração de nove características para teste: (1) informar o e-mail caso queira receber notícias das últimas 24h dos tópicos selecionados; (2) escolher a coleção de documentos; (3) escolha da representação de documentos e do algoritmo de classificação em (4); em (5) a opção de escolha dos tópicos de interesse; (6) a opção caso o usuário deseje visualizar as métricas de avaliação; na opção (7) salva o perfil para envio de e-mails diários com notícias de interesse do usuário assinante, além de mostra-las na tela (9), já na opção (8) são baixadas as últimas notícias (online) da coleção escolhida em (2) além de salvar o perfil para envio de e-mails diários com notícias de interesse do usuário assinante e mostra-las na tela (9).

**Figura 17 – Interface para testes e demonstrações do Sistema Recomendador de Notícias *Luppar News Recommender (Luppar News-Rec)***




Fonte: Elaborado pelo autor.

# 5 AVALIAÇÃO, RESULTADOS E DISCUSSÃO

O *Luppar News Recommender (**Luppar News-Rec**)* é um Sistema Recomendador de Notícias composto por algoritmos de classificação e representações de documentos que trabalham em conjunto para solucionar o problema de classificação de texto, no caso classificação de notícias, de forma a retornar as notícias que sejam relevantes aos usuários assinantes, com base em tópicos de interesse, e que apresente bom desempenho. Há ampla variedade de algoritmos e representações e assim coloca-se o problema de decidir uma ou mais combinações a utilizar.

Estes algoritmos de classificação e representação de documentos foram experimentados em diversas combinações com o objetivo compreender os efeitos dessas, dos seus parâmetros e das representações. No final, foram escolhidas as melhores combinações e essas implementadas no ***Luppar News-Rec***. Os experimentos são descritos a seguir.

## 5.1 SELEÇÃO DA REPRESENTAÇÃO-CLASSIFICADOR

Para a avaliação, os classificadores KNN - *K Nearest Neighbor*, SVM - *Support Vector Machines (Kernel: RBF)*, DT - *Decision Tree* e RF - *Random Forest* são combinados com as representações de documentos BoW, TF-IDF, *Word2Vec* e *FastText*, além da abordagem `E2V-IDF`, aplicada ao *Word2Vec* e *FastText*. A abordagem proposta `E2V-IDF` foi apresentada na seção 2.1.3.

Essas combinações são aplicadas às coleções de documentos próprias: **Z5News**, **Z5NewsBrasil** e **Z12News**, descritas no capítulo anterior. Além dessas, foram selecionadas quatro coleções de documentos em inglês já comumente usadas como referência na literatura de classificação de texto para avaliação das combinações propostas.

As coleções de documentos de referência (em inglês) são:

- *Reuters-21578 (R8)*;
- *Reuters-21578 (R52)*;
- *20 Newsgroups*[1];
- *Reuters (RCV1-Reuters Corpus Volume 1)*[2].

Todas essas coleções também encontram-se pré-processadas, ou seja, tokenizada, sem *stopwords* e com *stemming*. A Tabela 3 mostra as características dessas coleções.

---

[1] *Reuters-21578 (R8 / R52)* e 20News - Download: https://www.cs.umb.edu/~smimarog/textmining/datasets/

[2] *Reuters RCV1* - Download: http://www.daviddlewis.com/resources/testcollections/rcv1/ (utilizando apenas 33.149 devido a limitação computacional)

**Tabela 3 – Características das coleções de notícias de referência**

| Coleção | Idioma | Tipo | Notícias | Tópicos | Termos (média) |
|---|---|---|---|---|---|
| Reuters-21578 R8 | EN | Monorrótulo | 7.674 | 8 | 65 |
| Reuters-21578 R52 | EN | Monorrótulo | 9.100 | 52 | 69 |
| Reuters RCV1 | EN | Multirrótulo | 33.149 | 103 | 123 |
| 20 Newsgroups | EN | Monorrótulo | 18.821 | 20 | 141 |

Fonte: Elaborado pelo autor.

### 5.1.1 Testes preliminares

Alguns testes preliminares foram realizados objetivando a melhor parametrização dos algoritmos desenvolvidos, onde podemos destacar:

- Em relação ao pré-processamento, foram realizados testes utilizando Lematização (*lemmatization*) e Radicalização de palavras (*stemming*). Os resultados utilizando *stemming* foram superiores para a maioria das coleções, inclusive as coleções próprias;
- O classificador SVM utilizando *kernel* RBF e *kernel* Linear foram testados, e o *kernel* RBF obteve melhores resultados para representações *embeddings*;
- Variação dos parâmetros *gamma* e *C* para o classificador SVM *kernel* RBF, foram testados e os melhores resultados obtidos, se deram com os valores 0.01 e 1.0 respectivamente;
- Para o classificador KNN o valor de K, dentre os testados, que obteve os melhores resultados foi o K = 5 utilizando distância Euclidiana;
- Para os classificadores *Decision Tree* e *Random Forest* foram utilizados parâmetros padrões da biblioteca *Scikit-Learn*, com exceção do parâmetro: *min_samples_split* = 40, que foi baseado na literatura (UDACITY, 2016);
- Outros algoritmos foram testados, mas obtiveram resultados inferiores (*Logistic Regression*) e/ou apresentando longo tempo de processamento, como é o caso do MLP (*Multilayer Perceptron*). Esses algoritmos foram descartados para esse trabalho;
- Já em relação aos vetores *Embeddings*, foram testados 3 valores para o tamanho do vetor de característica (*size*), variando entre 100, 300 e 600 dimensões. A melhor relação tempo de processamento / resultados se deram com 300 dimensões. E em relação ao modelo de treinamento da janela de contexto, foram testadas os modelos *CBoW* e *Skip-gram*, e os melhores resultados foram obtidos pelo modelo *Skip-gram*;
- Testes Inter-bases foram realizados, ou seja, treinado o modelo com uma coleção de documento (R52) e testado com outra coleção (R8) e os resultados foram satisfatórios. Com base nisso, foi visto a viabilidade de treinamento de um modelo com dados de uma

fonte de dados, e que novas entradas (de outra fonte de dados) poderiam ser classificadas (preditas) com base nesse modelo. Maiores informações no APÊNDICE C.

### 5.1.2 Resultados da avaliação

As avaliações ocorreram em sequência combinando 7 coleções de documentos (*Reuters-21578 (R8), Reuters-21578 (R52), Reuters RCV1, 20 Newsgroups*, Z5News, Z5News Brasil e Z12News), 4 classificadores (SVM, KNN, DT e RF) e 6 representações de documentos (BoW, TF-IDF, *Word2Vec, Word2Vec* (`E2V-IDF`), *FastText* e *FastText* (`E2V-IDF`)), o que totaliza 168 execuções com métricas calculadas. Realizados em um sistema operacional Microsoft Windows (7 Professional) com processador *Intel Core i5* de 2,67 GHz com memória RAM de 16GB 1333 MHz DDR3. Todos as avaliações foram implementadas na linguagem *Python 3.7* e utilizando as bibliotecas da Tabela 4.

**Tabela 4 – Bibliotecas da linguagem Python utilizadas no projeto**

| **Biblioteca** | **Finalidade** | **Link** |
|---|---|---|
| Numpy | Programação matemática do projeto. | http://www.numpy.org/ |
| NLTK | Soluções que envolvem Processamento de Linguagem Natural na fase de pré-processamento. | https://www.nltk.org/ |
| Scikit-Learn | Biblioteca de aprendizado de máquina (Representação, Classificadores e Métricas). | http://scikit-learn.org/ |
| Gensim | Implementação do *Word2vec* e *FastText*. | https://radimrehurek.com/gensim/ |
| Matplotlib | biblioteca de plotagem (gráficos) | https://matplotlib.org/ |
| BeautifulSoup | biblioteca para *Web Crawler* | https://pypi.org/project/beautifulsoup4/ |

Fonte: Elaborado pelo autor.

A Tabela 5 apresenta os parâmetros utilizados nos diferentes algoritmos implementados em cada execução para esta pesquisa. A parametrização foi escolhida baseado na literatura e com possibilidades exequíveis para as coleções.

Além da Representação de Documentos *Embedding* treinado com base na coleção, também foi analisado o uso de vetores pré-treinados em inglês (Wikipedia[3]) e português (STIL[4]), que estão detalhados a seguir na Tabela 6.

[3] Disponível em: http://nilc.icmc.usp.br/embeddings
[4] Disponível em: https://github.com/lintseju/word_embedding

**Tabela 5 – Parametrização dos algoritmos utilizados durante os experimentos**

| Algoritmo | Função | Parâmetros |
|---|---|---|
| Preprocessor | Pré-processamento | Remove StopWords: stopword (nltk.corpus)<br>Radicalização: SnowballStemmer ( EN \| PT )<br>Remove Pontuação: punctuation (string)<br>Tokenização: word_tokenize (nltk) |
| SVM | Classificação | Kernel: RBF (*Radial Basis Function*)<br>gamma: 0.01<br>C: 1.0 |
| KNN | Classificação | n_neighbors: 5 (K)<br>p:2 (distancia euclidiana) |
| DT | Classificação | min_samples_split: 40 |
| RF | Classificação | min_samples_split: 40<br>n_estimators: 10<br>n_jobs: -1 |
| *Word2Vec* (W2V) | Representação de Documento *Embedding* (Treinado na coleção) | size: 300<br>window: 5<br>sg: 1 (*Skip-gram*) |
| *FastText* (FT) | Representação de Documento *Embedding* (Treinado na coleção) | size: 300<br>window: 5<br>sg: 1 (*Skip-gram*) |

Fonte: Elaborado pelo autor.

**Tabela 6 – Características dos vetores *embeddings* pré-treinados (Inglês e Português)**

| Nome | Ano | Idioma | Representação | Dimensões | Window | Modelo |
|---|---|---|---|---|---|---|
| Wikipedia | 2019 | EN | *Word2Vec* | 300 | 5 | *Skip-gram* |
| Wikipedia | 2019 | EN | *FastText* | 300 | 5 | *Skip-gram* |
| STIL | 2017 | PT | *Word2Vec* | 300 | 5 | *Skip-gram* |
| STIL | 2017 | PT | *FastText* | 300 | 5 | *Skip-gram* |

Fonte: Elaborado pelo autor.

Para cada um dos classificadores, foi aplicada a estratégia *One-Against-All* (Um-contra-todos) – que constrói $|C|$ modelos de predição binários, onde $|C|$ é o número de classes. Cada modelo é treinado para separar uma das classes das demais. Portanto, o i-ésimo modelo é treinado considerando que todos os exemplos da i-ésima classe pertencem à classe positiva, enquanto os exemplos das outras classes pertencem à classe negativa. Quando um exemplo de classe desconhecida é apresentado, ele é classificado por cada um dos $|C|$ modelos de predição e recebe a classe relativa ao modelo que obtiver o melhor resultado (HSU; LIN, 2002) quando a classificação é monorótulo. Para classificação multirótulo são escolhidas as classe de maiores escores gerados pelos classificadores.

Os treinamentos e testes utilizaram 10 iterações da validação cruzada (*10-fold*). Todos os resultados foram calculados pela micro-média da métrica: *f1-score (F1)* e também calculado a variância. Os valores em negrito indicam os três melhores resultados obtidos, para cada coleção, dentre as combinações.

A Tabela 7, mostra o desempenho alcançado por cada um dos classificadores aplicados às representações de documentos tradicionais: BoW e TF-IDF para as Coleções de Referência (R8, R52, RCV1 e 20News).

**Tabela 7 – Resultados da métrica *F1-Score* - Média(Variância) para coleções de Referência utilizando as representações: BoW e TF-IDF**

| Modelo | **R8** | **R52** | **RCV1** | **20 News** |
|---|---|---|---|---|
| *SVM(RBF)+BoW* | 0,9080 (0,0003) | **0,8675 (0,0003)** | 0,6845 (0,0005) | 0,5904 (0,0138) |
| *SVM(RBF)+TFIDF* | 0,8870 (0,0004) | 0,8573 (0,0004) | 0,5451 (0,0002) | **0,6674 (0,0161)** |
| *KNN+BoW* | 0,8965 (0,0003) | 0,8388 (0,0003) | 0,6989 (0,0002) | 0,2889 (0,0044) |
| *KNN+TFIDF* | 0,8795 (0,0005) | 0,8509 (0,0004) | **0,7672 (0,0002)** | 0,6397 (0,0089) |
| *DT+BoW* | 0,9101 (0,0003) | **0,8615 (0,0002)** | **0,7180 (0,0003)** | 0,5913 (0,0068) |
| *DT+TFIDF* | **0,9105 (0,0002)** | 0,8531 (0,0004) | **0,7220 (0,0001)** | 0,5760 (0,0082) |
| *RF+BoW* | **0,9208 (0,0003)** | 0,8504 (0,0004) | 0,6600 (0,0003) | **0,6716 (0,0112)** |
| *RF+TFIDF* | **0,9294 (0,0003)** | **0,8592 (0,0003)** | 0,6789 (0,0003) | **0,6621 (0,0107)** |

Fonte: Elaborado pelo autor.

Já a Tabela 8, mostra o desempenho alcançado por cada um dos classificadores aplicados às representações de documentos tradicionais: BoW e TF-IDF para as Coleções próprias (Z5News, Z5NewsBrasil e Z12News).

**Tabela 8 – Resultados da métrica *F1-Score* - Média(Variância) para coleções próprias utilizando as representações: BoW e TF-IDF**

| Modelo | **Z5News** | **Z5NewsBrasil** | **Z12News** |
|---|---|---|---|
| *SVM(RBF)+BoW* | **0,7873 (0,0008)** | **0,8025 (0,0007)** | **0,5907 (0,0399)** |
| *SVM(RBF)+TFIDF* | 0,7707 (0,0007) | **0,7744 (0,0006)** | **0,5637 (0,0410)** |
| *KNN+BoW* | 0,6957 (0,0001) | 0,6648 (0,0016) | 0,5039 (0,0219) |
| *KNN+TFIDF* | 0,7469 (0,0001) | 0,7469 (0,0016) | 0,5250 (0,0230) |
| *DT+BoW* | 0,7318 (0,0007) | 0,6825 (0,0037) | 0,4987 (0,0180) |
| *DT+TFIDF* | 0,7232 (0,0005) | 0,6667 (0,0039) | 0,4873 (0,0163) |
| *RF+BoW* | **0,7818 (0,0004)** | **0,7651 (0,0020)** | **0,5671 (0,0261)** |
| *RF+TFIDF* | **0,7781 (0,0004)** | 0,7589 (0,0026) | 0,5607 (0,0235) |

Fonte: Elaborado pelo autor.

Por outro lado, a Tabela 9 mostra o desempenho de cada classificador aplicados às representações de documentos *Embeddings (Word2Vec e FastText)*, treinados com base nos termos da coleção, para as coleções de Referência (R8, R52, RCV1 e 20News).

**Tabela 9 – Resultados da métrica *F1-Score* para coleções de referência utilizando as representações *embeddings*: *Word2Vec* e *FastText* - Treinadas na Coleção**

| Modelo | R8 | R52 | RCV1 | 20 News |
|---|---|---|---|---|
| *SVM(RBF)+W2V* | 0,9239 (0,00030) | 0,8486 (0,0006) | 0,6451 (0,0002) | 0,5065 (0,0377) |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,9608 (0,00005)** | **0,9107 (0,0003)** | **0,7821 (0,0002)** | **0,6955 (0,0134)** |
| *SVM(RBF)+FT* | 0,9188 (0,00030) | 0,8433 (0,0007) | 0,6360 (0,0003) | 0,4957 (0,0313) |
| *SVM(RBF)+FT* (E2V-IDF) | **0,9616 (0,00004)** | **0,9091 (0,0002)** | 0,7735 (0,0002) | **0,6705 (0,0140)** |
| *KNN+W2V* | **0,9599 (0,00010)** | 0,9064 (0,0003) | **0,7905 (0,0002)** | 0,6283 (0,0140) |
| *KNN+W2V* (E2V-IDF) | 0,9562 (0,00010) | 0,9064 (0,0004) | 0,7746 (0,0003) | **0,6383 (0,0137)** |
| *KNN+FT* | 0,9593 (0,00010) | 0,9058 (0,0003) | **0,7859 (0,0003)** | 0,6041 (0,0119) |
| *KNN+FT* (E2V-IDF) | 0,9567 (0,00005) | **0,9070 (0,0003)** | 0,7661 (0,0003) | 0,6104 (0,0125) |
| *DT+W2V* | 0,9106 (0,00020) | 0,8079 (0,0006) | 0,6351 (0,0002) | 0,3948 (0,0072) |
| *DT+W2V* (E2V-IDF) | 0,9152 (0,00030) | 0,8098 (0,0008) | 0,6241 (0,0002) | 0,4156 (0,0068) |
| *DT+FT* | 0,9154 (0,00010) | 0,8037 (0,0006) | 0,6292 (0,0002) | 0,3489 (0,0053) |
| *DT+FT* (E2V-IDF) | 0,9088 (0,00030) | 0,8080 (0,0006) | 0,6078 (0,0002) | 0,3707 (0,0056) |
| *RF+W2V* | 0,9522 (0,00020) | 0,8767 (0,0005) | 0,6985 (0,0002) | 0,5156 (0,0136) |
| *RF+W2V* (E2V-IDF) | 0,9467 (0,00010) | 0,8787 (0,0005) | 0,6870 (0,0002) | 0,5337 (0,0126) |
| *RF+FT* | 0,9498 (0,00010) | 0,8770 (0,0005) | 0,6904 (0,0002) | 0,4743 (0,0114) |
| *RF+FT* (E2V-IDF) | 0,9493 (0,00010) | 0,8744 (0,0004) | 0,6716 (0,0002) | 0,4896 (0,0112) |

Fonte: Elaborado pelo autor.

Na Tabela 10 mostra o desempenho de cada um dos classificadores aplicados às representações de documentos *Embeddings (Word2Vec e FastText)*, treinados com base nos termos da coleção, para as coleções próprias (Z5News, Z5NewsBrasil e Z12News).

**Tabela 10 – Resultados da métrica *F1-Score* para coleções próprias utilizando as representações: *Word2Vec* e *FastText*- Treinadas na Coleção**

| Modelo | Z5News | Z5NewsBrasil | Z12News |
|---|---|---|---|
| *SVM(RBF)+W2V* | 0,7734 (0,0004) | **0,7742 (0,0004)** | 0,5321 (0,0417) |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,7838 (0,0003)** | **0,7920 (0,0005)** | **0,5954 (0,0317)** |
| *SVM(RBF)+FT* | **0,7738 (0,0006)** | 0,7712 (0,0004) | 0,5267 (0,0412) |
| *SVM(RBF)+FT* (E2V-IDF) | **0,7846 (0,0004)** | **0,7865 (0,0005)** | **0,5828 (0,0412)** |
| *KNN+W2V* | 0,7648 (0,0002) | 0,7509 (0,0010) | 0,5436 (0,0236) |
| *KNN+W2V* (E2V-IDF) | 0,7602 (0,0002) | 0,7533 (0,0009) | 0,5392 (0,0246) |
| *KNN+FT* | 0,7642 (0,0002) | 0,7500 (0,0012) | **0,5503 (0,0165)** |
| *KNN+FT* (E2V-IDF) | 0,7620 (0,0002) | 0,7493 (0,0011) | 0,5406 (0,0239) |
| *DT+W2V* | 0,6930 (0,0004) | 0,6742 (0,0027) | 0,4074 (0,0100) |
| *DT+W2V* (E2V-IDF) | 0,6888 (0,0003) | 0,6788 (0,0024) | 0,4107 (0,0102) |
| *DT+FT* | 0,6920 (0,0004) | 0,6673 (0,0027) | 0,4020 (0,0101) |
| *DT+FT* (E2V-IDF) | 0,6770 (0,0003) | 0,6684 (0,0031) | 0,4047 (0,0101) |
| *RF+W2V* | 0,7612 (0,0003) | 0,7515 (0,0013) | 0,5190 (0,0209) |
| *RF+W2V* (E2V-IDF) | 0,7562 (0,0003) | 0,7528 (0,0010) | 0,5224 (0,0210) |
| *RF+FT* | 0,7592 (0,0004) | 0,7476 (0,0015) | 0,5182 (0,0210) |
| *RF+FT* (E2V-IDF) | 0,7518 (0,0004) | 0,7499 (0,0016) | 0,5186 (0,0210) |

Fonte: Elaborado pelo autor.

A Tabela 11 apresenta o desempenho de cada classificador aplicados às representações de documentos *Embeddings (Word2Vec e FastText)* utilizando vetor *embedding* pré-treinado com base nos termos da *wikipedia* (2019 - EN), para as coleções de Referência.

**Tabela 11 – Resultados da métrica *F1-Score* para coleções de referência utilizando as representações *embeddings*: *Word2Vec* e *FastText* - Pré-Treinadas (Wiki)**

| Modelo | R8 | R52 | RCV1 | 20 News |
|---|---|---|---|---|
| *SVM(RBF)+W2V* | 0,8359 (0,0014) | 0,7731 (0,0010) | 0,4972 (0,0003) | 0,4831 (0,0323) |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,9200 (0,0003)** | **0,8705 (0,0001)** | 0,6937 (0,0002) | **0,5838 (0,0239)** |
| *SVM(RBF)+FT* | 0,8361 (0,0015) | 0,7674 (0,0011) | 0,5019 (0,0002) | 0,4822 (0,0331) |
| *SVM(RBF)+FT* (E2V-IDF) | **0,9279 (0,0003)** | **0,8759 (0,0001)** | 0,6969 (0,0002) | **0,5859 (0,0256)** |
| *KNN+W2V* | 0,9196 (0,0002) | 0,8586 (0,0004) | **0,7416 (0,0003)** | 0,5036 (0,0167) |
| *KNN+W2V* (E2V-IDF) | 0,8716 (0,0004) | 0,8133 (0,0003) | 0,7004 (0,0004) | **0,5261 (0,0191)** |
| *KNN+FT* | **0,9251 (0,0002)** | **0,8626 (0,0004)** | **0,7470 (0,0003)** | 0,5086 (0,0176) |
| *KNN+FT* (E2V-IDF) | 0,8787 (0,0003) | 0,8234 (0,0003) | **0,7070 (0,0005)** | 0,5233 (0,0211) |
| *DT+W2V* | 0,7948 (0,0011) | 0,6844 (0,0012) | 0,5414 (0,0001) | 0,2539 (0,0046) |
| *DT+W2V* (E2V-IDF) | 0,7365 (0,0007) | 0,6167 (0,0017) | 0,4985 (0,0002) | 0,2579 (0,0060) |
| *DT+FT* | 0,8047 (0,0006) | 0,6882 (0,0013) | 0,5457 (0,0001) | 0,2684 (0,0049) |
| *DT+FT* (E2V-IDF) | 0,7533 (0,0009) | 0,6320 (0,0011) | 0,5081 (0,0001) | 0,2702 (0,0057) |
| *RF+W2V* | 0,8710 (0,0004) | 0,7747 (0,0009) | 0,5903 (0,0001) | 0,3498 (0,0120) |
| *RF+W2V* (E2V-IDF) | 0,8249 (0,0007) | 0,7284 (0,0011) | 0,5427 (0,0002) | 0,3625 (0,0145) |
| *RF+FT* | 0,8769 (0,0006) | 0,7848 (0,0009) | 0,6011 (0,0001) | 0,3683 (0,0140) |
| *RF+FT* (E2V-IDF) | 0,8349 (0,0009) | 0,7367 (0,0008) | 0,5589 (0,0002) | 0,3727 (0,0160) |

Fonte: Elaborado pelo autor.

Já na Tabela 12, os resultados obtidos utilizando vetor *embedding* pré-treinado com base nos termos da *wikipedia* (2019 - EN) para as coleções próprias em Inglês (Z5News e Z12News) e o vetor *embedding* pré-treinado STIL (2017 - PT) para a coleção própria em Português (Z5NewsBrasil), aplicados aos classificadores estudados.

**Tabela 12 – Apresenta o *F1-Score* para coleções próprias utilizando representações *embedding*: *Word2Vec* e *FastText* - Pré-Treinado (Wiki + STIL)**

| Modelo | Z5News | Z5NewsBrasil | Z12News |
|---|---|---|---|
| *SVM(RBF)+W2V* | 0,7553 (0,0006) | 0,6679 (0,0010) | 0,4989 (0,0366) |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,7624 (0,0005)** | **0,7111 (0,0005)** | **0,5429 (0,0394)** |
| *SVM(RBF)+FT* | **0,7556 (0,0007)** | **0,7283 (0,0005)** | 0,5023 (0,0370) |
| *SVM(RBF)+FT* (E2V-IDF) | **0,7641 (0,0005)** | **0,7501 (0,0008)** | **0,5468 (0,0397)** |
| *KNN+W2V* | 0,7291 (0,0003) | 0,5908 (0,0016) | 0,4897 (0,0187) |
| *KNN+W2V* (E2V-IDF) | 0,7055 (0,0002) | 0,5986 (0,0017) | 0,4755 (0,0180) |
| *KNN+FT* | 0,7353 (0,0003) | 0,6789 (0,0008) | **0,5040 (0,0129)** |
| *KNN+FT* (E2V-IDF) | 0,7143 (0,0003) | 0,6771 (0,0011) | 0,4900 (0,0132) |
| *DT+W2V* | 0,5836 (0,0005) | 0,4151 (0,0016) | 0,3167 (0,0057) |
| *DT+W2V* (E2V-IDF) | 0,5550 (0,0003) | 0,4093 (0,0015) | 0,3021 (0,0047) |
| *DT+FT* | 0,5996 (0,0004) | 0,4609 (0,0018) | 0,3224 (0,0054) |
| *DT+FT* (E2V-IDF) | 0,5673 (0,0003) | 0,4463 (0,0011) | 0,3105 (0,0057) |
| *RF+W2V* | 0,7037 (0,0004) | 0,5537 (0,0014) | 0,4453 (0,0165) |
| *RF+W2V* (E2V-IDF) | 0,6771 (0,0006) | 0,5520 (0,0017) | 0,4338 (0,0154) |
| *RF+FT* | 0,7107 (0,0004) | 0,6063 (0,0015) | 0,4536 (0,0171) |
| *RF+FT* (E2V-IDF) | 0,6823 (0,0005) | 0,5963 (0,0017) | 0,4409 (0,0163) |

Fonte: Elaborado pelo autor.

### 5.1.3 Discussão e Seleção

A primeira análise foi sobre a necessidade de testes estatísticos para validar as discriminações entre as combinações representação-classificador. Observou-se que eles eram desnecessários devido as baixas variâncias em relação às diferenças das médias entre as combinações do 'primeiro pelotão' e as que vinham em seguida. Isso também foi facilitado pelo fato de que não se iria escolher apenas uma combinação, mas o melhor grupo de três delas.

É válido observar algumas características das coleções de documentos utilizadas nos experimentos realizados nesse trabalho, conforme os resultados das tabelas 2 e 3. As coleções de documentos podem ser categorizadas em Curtas, Médias e Longas. Curtas (Até 22 termos - Z5News e Z5NewsBrasil), obteve os melhores resultados combinando os classificadores com representações tradicionais: BoW e TF-IDF. Médias (Até 69 termos - R8, R2 e Z12News), obtiveram os melhores resultados utilizando representações *Embeddings*. Longas (Até 141 termos - 20 Newsgroups e RCV1), os melhores resultados também foram obtidos utilizando representações *Embeddings*.

Observe alguns exemplos de resultados (*F1-Score*) das tabelas 8 e 10, onde o fato de utilizar representações *Embeddings* na coleção de documentos própria Z5NewsBrasil (em Português), que teve seu vetor *embedding* treinado com base na coleção, não superou o tradicional BoW, mas seguido de representações de documentos *embeddings* ponderadas por IDF. Para a também coleção própria Z5News (em Inglês) o melhor resultado foi utilizando a representação BoW, mas seguido também de representações *Embeddings*. Além disso podemos também observar que a variância dos resultados dos *embeddings* são menores do que das representações tradicionais. Já em relação a coleção própria Z12News (Inglês) os melhores resultados foram obtidos por representações *embeddings*.

Note que ainda em relação a coleção própria Z5News, na tabela 13, são comparadas as três combinações mais bem avaliadas, uma delas utilizando a representação tradicional BoW e as demais utilizando representações *Embeddings (Word2Vec/FastText)*, vemos que os resultados da métrica *Recall* foram maiores para as representações *Embeddings* e o número de documentos retornados (classificados) também são maiores.

No que se diz respeito a abordagem *Embedding* `E2V-IDF` (ponderada por IDF) podemos observar nas tabelas: 9, 10, 11 e 12, que a abordagem quando combinada com o classificador SVM (*kernel* RBF) apresenta resultados melhores, chegando a uma melhoria de até 19% (20 Newsgroups), em relação as abordagens de representações *Embeddings* tradicionais

(por média) tanto para o *Word2Vec* quanto para o *FastText* em todas as coleções de documentos estudadas, tanto para os vetores *embeddings* treinados com base na coleção, quanto para vetores pré-treinados. Já quando combinada com os demais classificadores, não se obtém melhorias significativas para a grande maioria das combinações aplicadas à todas as coleções estudadas.

Outro ponto é o fato de que os resultados das combinações que utilizam vetores *embeddings* treinados com base nos dados da coleção, obtiveram resultados superiores para todas as coleções e combinações, em relação aos vetores *embeddings* pré-treinados.

#### 5.1.3.1 Seleção

As combinações mais bem avaliadas foram as seguintes respectivamente: ***SVM(RBF) +FT*** (`E2V-IDF`), ***SVM(RBF)+W2V*** (`E2V-IDF`) e ***RF+BoW***. Ou seja, os classificadores SVM *kernel:* RBF e *Random Forest* (RT) combinados com as representações *embeddings*: *FastText e Word2Vec*, ambas aplicando a abordagem `E2V-IDF`, além da tradicional representação BoW (*Bag-of-Words*), obtiveram os melhores resultados para a maioria das coleções de documentos avaliadas. Esses classificadores e representações de documentos, além das coleções Z5News, Z5NewsBrasil e Z12News, foram implementados no ***Luppar News-Rec***.

Analisando individualmente as três combinações classificador/representação de documentos mais bem avaliadas, *SVM(RBF)+FT* (`E2V-IDF`), *SVM(RBF)+W2V* (`E2V-IDF`) e *RF+BoW* para a coleção de documento própria (Z5News), agora não mais utilizando *cross-validation*, mas sim, a divisão da coleção de documento em 80% para dados de treinamento e 20% para teste. Dentre as combinações avaliadas, a que obteve os melhores resultados para a métrica: *recall* e com isso retornando mais notícias, foi a combinação: *SVM(RBF)+FT* (`E2V-IDF`), como podemos ver na Tabela 13:

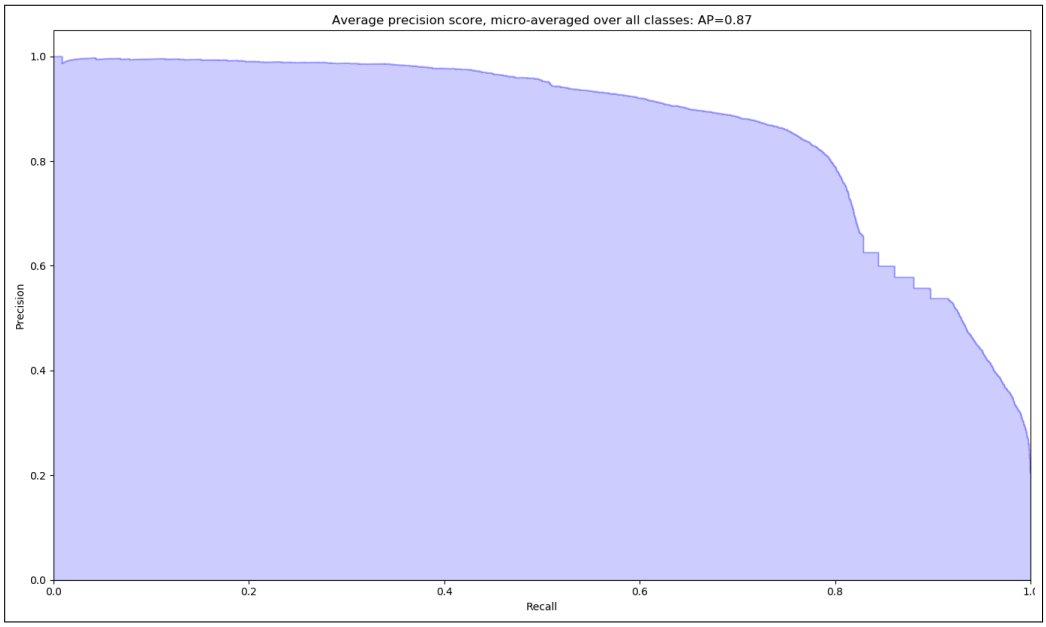
**Tabela 13 – Comparativo das combinações mais bem avaliadas**

| Coleção Z5News | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Tam.Test** | **Retornados** |
| *RF+BoW* | 0,8964 | 0,6734 | 0,7691 | 0,6552 | 8507 | 6391 |
| *SVM(RBF)+FT* (`E2V-IDF`) | **0,8673** | **0,7369** | **0.7968** | **0,7267** | **8507** | **7228** |
| *SVM(RBF)+W2V* (`E2V-IDF`) | 0,8685 | 0,7361 | 0.7968 | 0,7268 | 8507 | 7210 |

Fonte: Elaborado pelo autor.

A seguir é detalhado graficamente os resultados obtidos com a combinação: *SVM (RBF) +FT* (`E2V-IDF`), onde alcançou-se uma *Average Precision* de 87% para todos os tópicos, que corresponde a Área sob a Curva *Precision-Recall* (PR-AUC), apresentada na Figura 18.

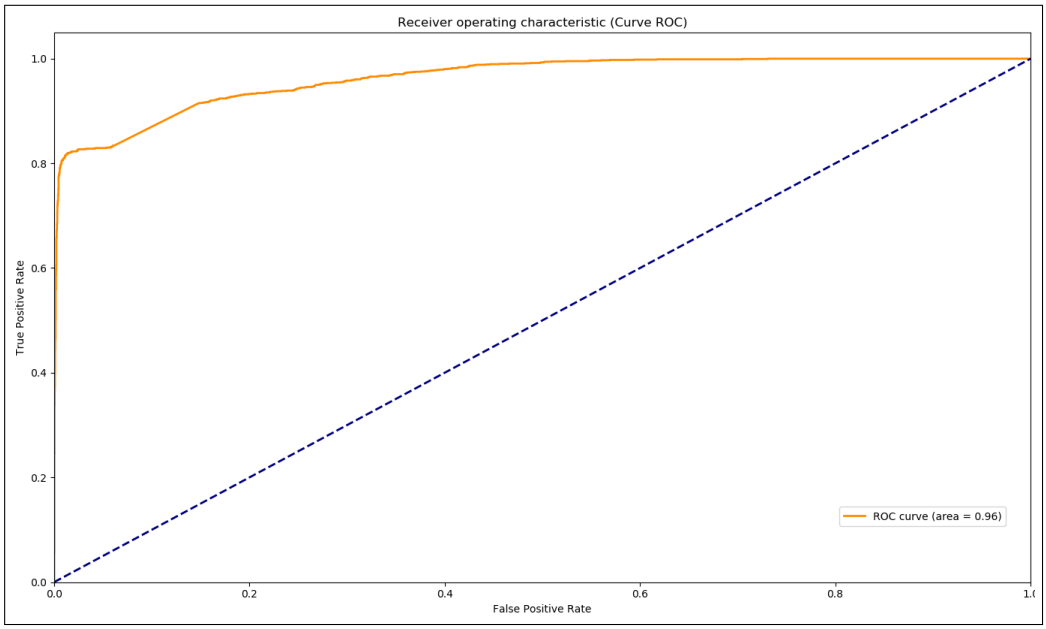
**Figura 18 – Área sob a Curva *Precision-Recall* com *Average Precision* de 87% para todos os tópicos da coleção Z5News (combinação: SVM(RBF)+FT (E2V-IDF))**




Fonte: Elaborado pelo autor.

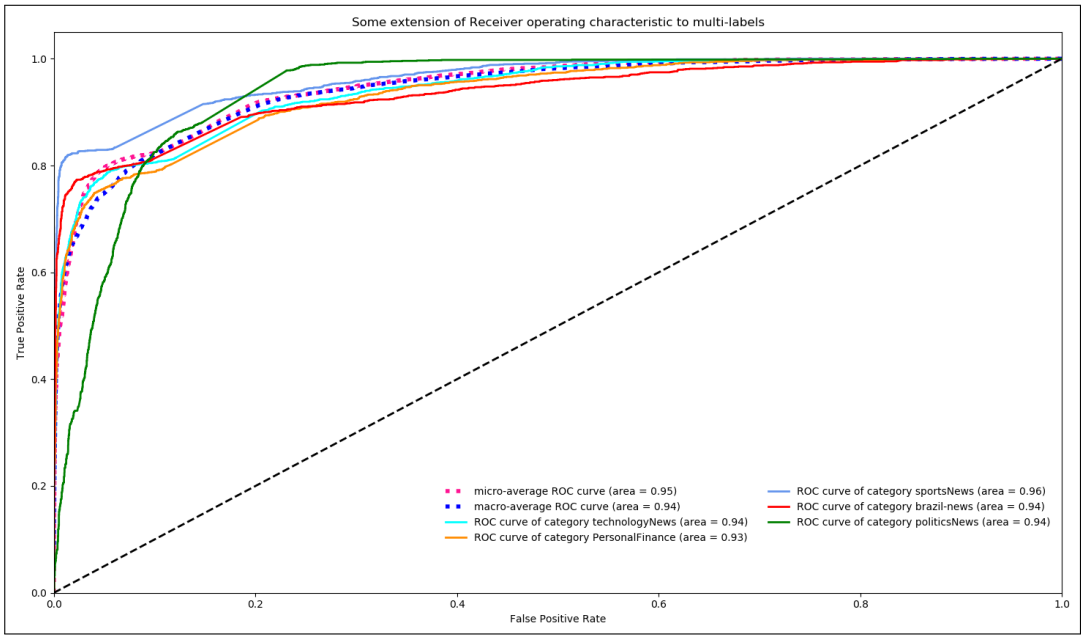
A Figura 19 mostra a curva ROC (*Receiver Operating Characteristic*) média enquanto a Figura 20 a detalhada. Na Figura 19, a Curva ROC para todos os tópicos confirma o que já estava revelado na curva *precision-recall* (Figura 18). A Figura 20 mostra um detalhamento por tópico, apresentando um desempenho satisfatório em praticamente todos os tópicos.

**Figura 19 – Curva ROC média da coleção Z5News (*SVM(RBF)+FT* (`E2V-IDF`))**




Fonte: Elaborado pelo autor.

**Figura 20 – Curva ROC detalhada por tópico para a coleção Z5News (combinação: *SVM(RBF)+FT* (E2V-IDF))**




Fonte: Elaborado pelo autor.

## 5.2 PLANO EXPERIMENTAL DE TESTES DO SISTEMA E MÉTRICAS

O ***Luppar News-Rec*** foi testado tendo como base as combinações classificador/representação de documentos mais bem avaliadas durante a fase de testes e validações de parâmetros (seção: 5.1), e aplicados às coleções de documentos próprias. A seguir descreve-se os recursos utilizados durante essa fase. Aqueles marcados com asterisco (*) não foram implementados no ***Luppar News-Rec*** - versão disponível na web.

- Coleções de Documentos
  - Z5News;
  - Z5NewsBrasil;
  - Z12News;
  - Z20News*.
- Representações de Documentos
  - BoW;
  - FastText + (E2V-IDF);
  - Word2Vec + (E2V-IDF).
- Classificadores
  - SVM e RF (*Random Forest*).

Foram realizadas duas etapas de testes para análise de desempenho do SRN desenvolvido. Esses testes estão detalhados na Tabela 2.

- **Testes *Offline***
  - Contendo 769 notícias curtas em Inglês (Z5News);
  - Contendo 656 notícias curtas em Português (Z5NewsBrasil);
  - Contendo 2246 notícias curtas em Inglês (Z20News);
  - Contendo 1360 notícias curtas em Inglês (Z12News).
- **Testes *Online***
  - Contendo 187 notícias em inglês (Z5News);
  - Contendo 189 notícias em Português (Z5NewsBrasil);
  - Contendo 376 notícias curtas em Inglês (Z20News);
  - Contendo 232 notícias curtas em Inglês (Z12News).

Esses testes foram aplicados aos modelos gerados pelas combinações oferecidas pelo ***Luppar News-Rec***. As métricas utilizadas para análise dos resultados serão as seguintes: *Precision*, *Recall*, *F1-Score*, Acurácia, RMSE, Curva UAC e Curva ROC, já descritas. Os resultados serão apresentados a seguir.

Além desses testes, foi aplicado o chamado **Teste *Online*: Avaliação por Tópico e por Assinante**, onde foi criada uma Matriz de Confusão detalhada por tópicos e suas respectivas métricas, além de simular e avaliar a inscrição de assinantes fictícios à tópicos aleatórios.

## 5.3 RESULTADOS

Os resultados dos testes aplicados ao ***Luppar News-Rec*** são detalhados nessa seção e na seção seguinte eles são discutidos.

### 5.3.1 Testes *Offline*

Esta subseção apresenta os resultados obtidos aplicando os Testes *Offline* (notícias de 3 à 10 dias, dependendo do tópico) às combinações oferecidas pelo ***Luppar News-Rec***, nas coleções de documentos: Z5News, Z5NewsBrasil, Z12News e Z20News.

As tabelas 14 e 15 mostram os resultados obtidos para as coleções de documentos Z5News e Z5NewsBrasil, respectivamente. Pode-se observar que as combinações envolvendo *embeddings* obtiveram os melhores resultados para ambas as coleções (*F1-Score* e Acurácia).

**Tabela 14 – Comparativo de combinações para a Coleção Z5News (*Offline*)**

| Coleção Z5News (*Offline*) contendo 769 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,8710 | 0,5969 | 0,7083 | 0,5721 | 527 | 0,3135 |
| *SVM(RBF)+FT* (E2V-IDF) | **0,8349** | **0,6970** | **0,7597** | **0,6879** | **642** | **0,2839** |
| *SVM(RBF)+W2V* (E2V-IDF) | 0,8273 | 0,6853 | 0,7496 | 0,6762 | 637 | 0,2835 |

Fonte: Elaborado pelo autor.

**Tabela 15 – Comparativo de combinações para a Coleção Z5NewsBrasil (*Offline*)**

| Coleção Z5NewsBrasil (*Offline*) contendo 656 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,8043 | 0,6326 | 0,7182 | 0,5976 | 516 | 0,3229 |
| *SVM(RBF)+FT* (E2V-IDF) | 0,8308 | 0,6738 | 0,7441 | 0,6738 | 532 | 0,3044 |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,8259** | **0,7012** | **0,7585** | **0,6951** | **557** | **0,2989** |

Fonte: Elaborado pelo autor.

Já as tabelas 16 e 17 mostram os resultados obtidos para as coleções de documentos Z12News e Z20News, respectivamente. Observa-se que a combinação: SVM(RBF)+W2V (E2V-IDF) obteve os melhores resultados para ambas as coleções (Acurácia / Documentos Retornados).

**Tabela 16 – Comparativo de combinações para a Coleção Z12News (*Offline*)**

| Coleção Z12News (*Offline*) contendo 1360 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,7800 | 0,3390 | 0,4726 | 0,3316 | 591 | 0,2511 |
| *SVM(RBF)+FT* (E2V-IDF) | 0,7967 | 0,5044 | 0,6177 | 0,5000 | 861 | 0,2281 |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,7758** | **0,5088** | **0,6146** | **0,5029** | **892** | **0,2306** |

Fonte: Elaborado pelo autor.

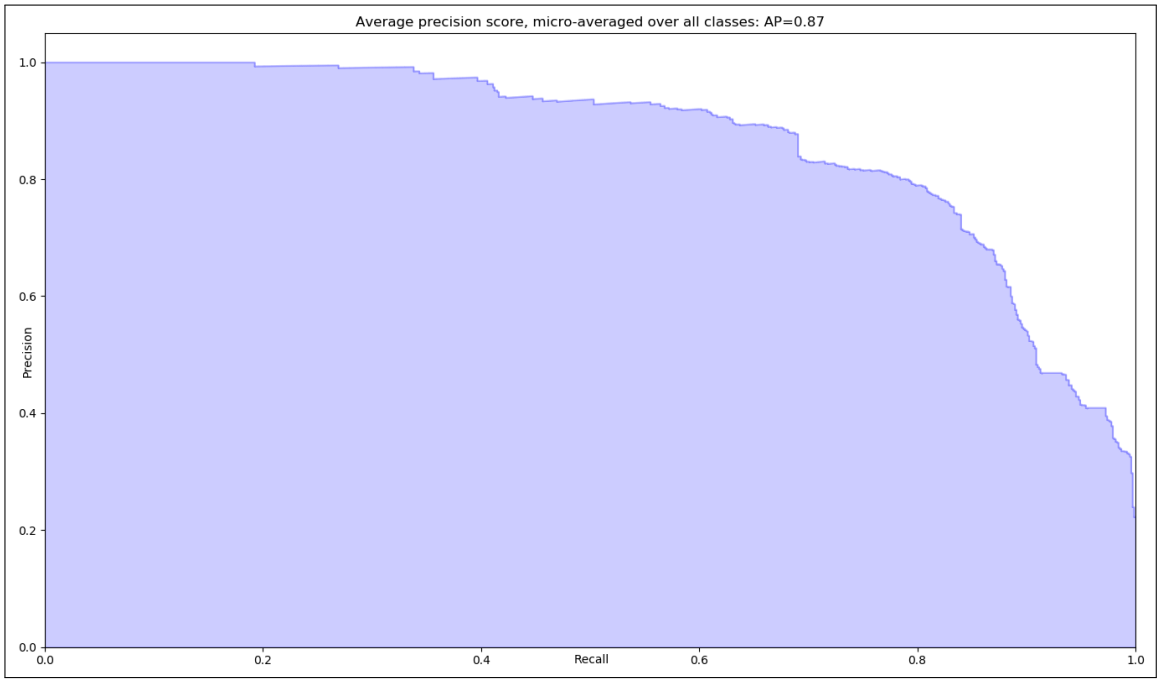
**Tabela 17 – Comparativo de combinações para a Coleção Z20News (*Offline*)**

| Coleção Z20News (*Offline*) contendo 2246 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,6344 | 0,1723 | 0,2719 | 0,1678 | 610 | 0,2153 |
| *SVM(RBF)+FT* (E2V-IDF) | 0,7174 | 0,3210 | 0,4436 | 0,3201 | 1005 | 0,2007 |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,7071** | **0,3246** | **0,4449** | **0,3232** | **1031** | **0,2012** |

Fonte: Elaborado pelo autor.

Abaixo estes resultados são detalhados de forma gráfica mostrando os melhores resultados (*F1-Score*) obtidos nos Testes *Offline*, ou seja, utilizando a coleção de documento Z5News, aplicando as 769 notícias de teste ao modelo obtido pela combinação (SVM(RBF)+FT (E2V-IDF)). Obteve-se uma *Average Precision* de 87% para todos os tópicos, que corresponde a Área sob a Curva *Precision-Recall* (PR-AUC), apresentada na Figura 21.

**Figura 21 – Área sob a Curva *Precision-Recall* com *Average Precision* de 87% para todos os tópicos da coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))**

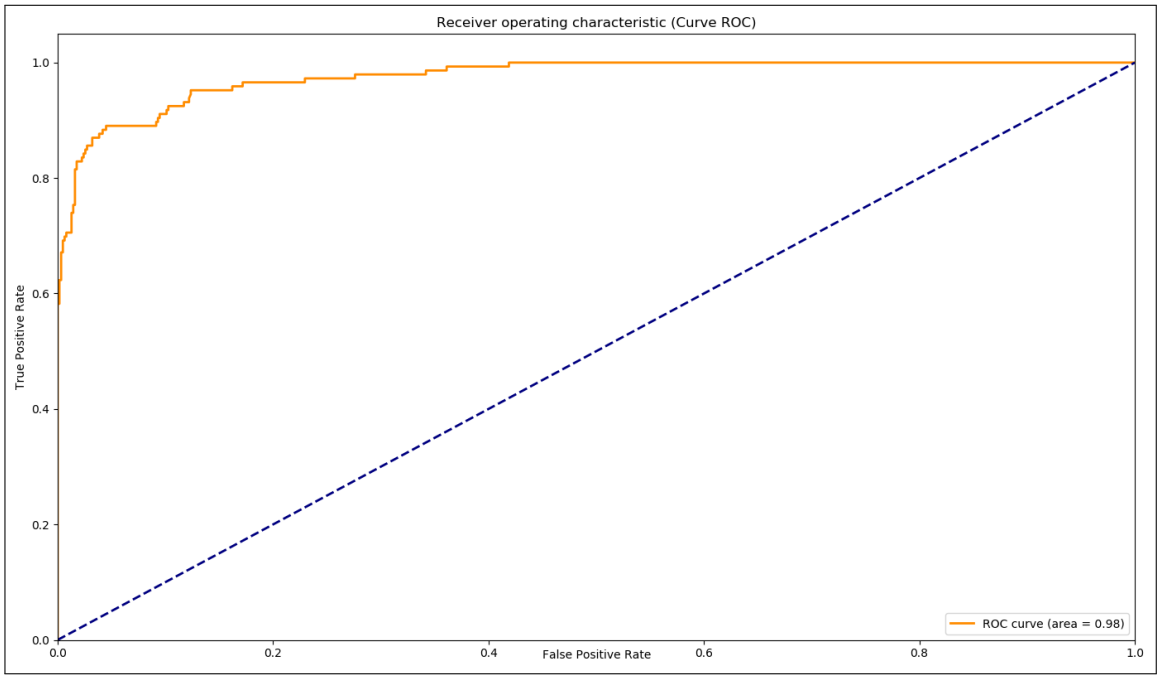



Fonte: Elaborado pelo autor.

A Figura 22 mostra a curva ROC média enquanto na Figura 23 é mostrado a curva ROC detalhada. Na Figura 22, a Curva ROC média para todos os tópicos confirma o que já estava revelado na curva *precision-recall* (Figura 21).

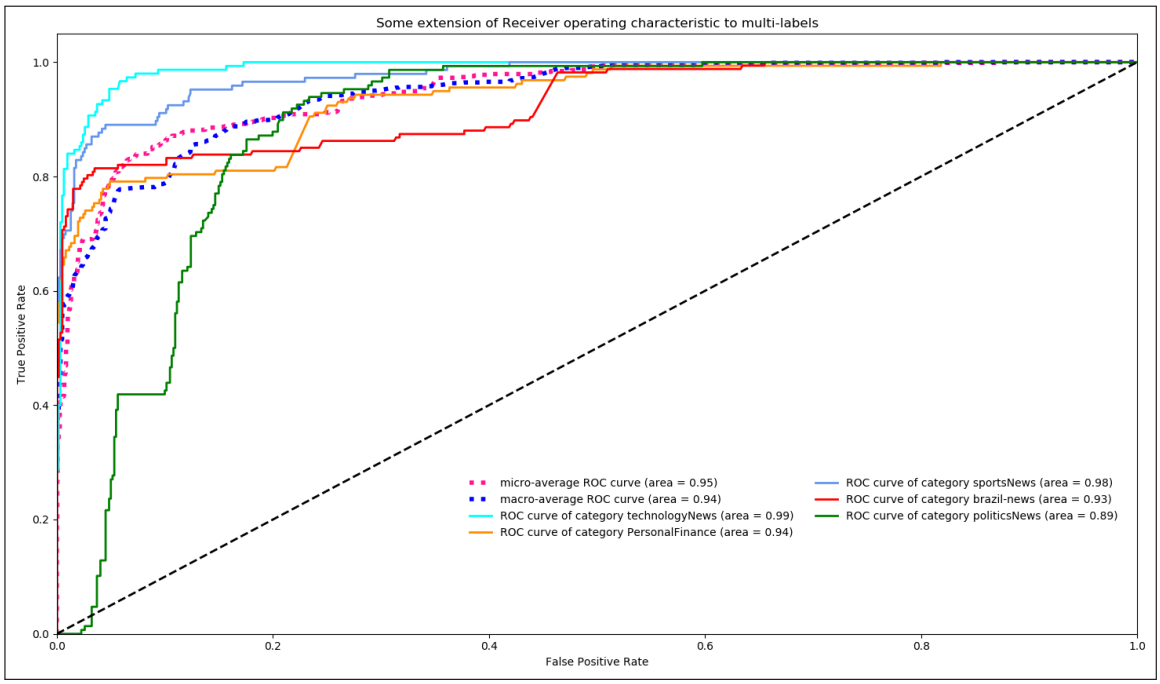
Já na Figura 23, que mostra um detalhamento por tópico, pode-se ver que a combinação tem desempenho satisfatório em praticamente todos os tópicos.

**Figura 22 – Curva ROC média da coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))**




Fonte: Elaborado pelo autor.

**Figura 23 – Curva ROC detalhada por tópico para a coleção Z5News (*Offline*) (combinação: SVM(RBF) +FT (E2V-IDF))**




Fonte: Elaborado pelo autor.

### 5.3.2 Testes *Online*

Nessa subseção são apresentados os resultados obtidos aplicando os Testes *Online* (notícias das últimas 24hs - coletadas em 20 de Agosto de 2019) às combinações oferecidas pelo ***Luppar News-Rec***, nas coleções de documentos: Z5News, Z5NewsBrasil, Z12News e Z20News.

As tabelas 18 e 19 mostram os resultados obtidos pelas combinações oferecidas pelo ***LupparRecNews*** aplicadas as coleções de documentos Z5News e Z5NewsBrasil respectivamente para notícias de 1 dia. Observar que a combinação: SVM(RBF)+FT (`E2V-IDF`) também obteve os melhores resultados para ambas as coleções (*F1-Score*) e menor RMSE.

**Tabela 18 – Comparativo de combinações para a Coleção Z5News (*Online*)**

| Coleção Z5News (*Online*) contendo 187 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,8392 | 0,6417 | 0,7273 | 0,6364 | 143 | 0,3103 |
| *SVM(RBF)+FT* (`E2V-IDF`) | **0,8951** | **0,7754** | **0,8309** | **0,7754** | **162** | **0,2512** |
| *SVM(RBF)+W2V* (`E2V-IDF`) | 0,8846 | 0,7380 | 0,8047 | 0,7380 | 156 | 0,2677 |

Fonte: Elaborado pelo autor.

**Tabela 19 – Comparativo de combinações para a Coleção Z5NewsBrasil (*Online*)**

| Coleção Z5NewsBrasil (*Online*) contendo 189 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,7566 | 0,6085 | 0,6745 | 0,5608 | 152 | 0,3427 |
| *SVM(RBF)+FT* (`E2V-IDF`) | **0,8125** | **0,6878** | **0,7450** | **0,6825** | **160** | **0,3069** |
| *SVM(RBF)+W2V* (`E2V-IDF`) | 0,8000 | 0,6772 | 0,7335 | 0,6772 | 160 | 0,3137 |

Fonte: Elaborado pelo autor.

As tabelas 20 e 21 mostram os resultados obtidos para as coleções de documentos: Z12News e Z20News respectivamente. Note-se que a combinação: SVM(RBF)+W2V (`E2V-IDF`) obteve os melhores resultados para ambas as coleções (*F1-Score* e Acurácia).

**Tabela 20 – Comparativo de combinações para a Coleção Z12News (*Online*)**

| Coleção Z12News (*Online*) contendo 232 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,7800 | 0,3362 | 0,4699 | 0,3362 | 100 | 0,2514 |
| *SVM(RBF)+FT* (`E2V-IDF`) | 0,8723 | 0,5302 | 0,6595 | 0,5302 | 141 | 0,2136 |
| *SVM(RBF)+W2V* (`E2V-IDF`) | **0,8562** | **0,5388** | **0,6614** | **0,5345** | **146** | **0,2144** |

Fonte: Elaborado pelo autor.

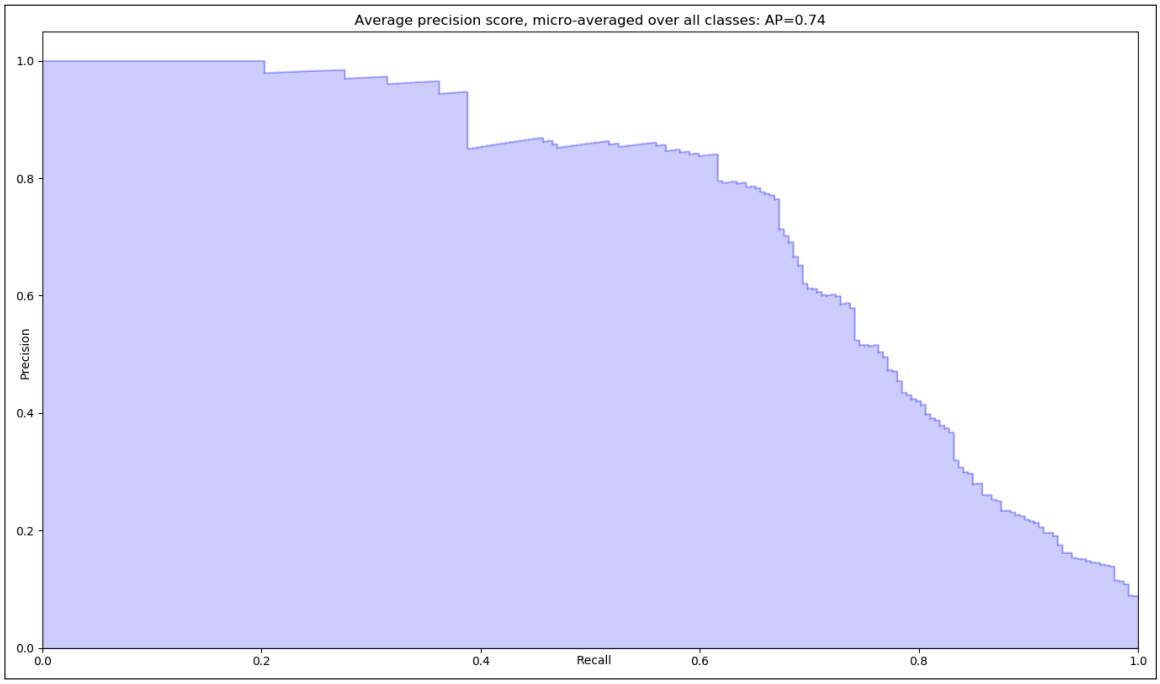
**Tabela 21 – Comparativo de combinações para a Coleção Z20News (*Online*)**

| Coleção Z20News (*Online*) contendo 376 notícias para teste | | | | | | |
|---|---|---|---|---|---|---|
| Modelo | **Precision** | **Recall** | **F1-Score** | **Acurácia** | **Retornados** | **RMSE** |
| *RF+BoW* | 0,7033 | 0,1702 | 0,2741 | 0,1702 | 91 | 0,2123 |
| *SVM(RBF)+FT* (E2V-IDF) | 0,7299 | 0,3378 | 0,4618 | 0,3351 | 174 | 0,1984 |
| *SVM(RBF)+W2V* (E2V-IDF) | **0,7345** | **0,3457** | **0,4702** | **0,3431** | **177** | **0,1974** |

Fonte: Elaborado pelo autor.

Em seguida, detalha-se graficamente os resultados obtidos pela coleção de documentos Z12News para os Testes *Online*, aplicando as 232 notícias ao modelo obtido pela combinação (SVM(RBF)+W2V (E2V-IDF)). Alcançou-se uma *Average Precision* de 74% para todas os tópicos, que corresponde a Área sob a Curva *Precision-Recall* (PR-AUC), apresentada abaixo na Figura 24.

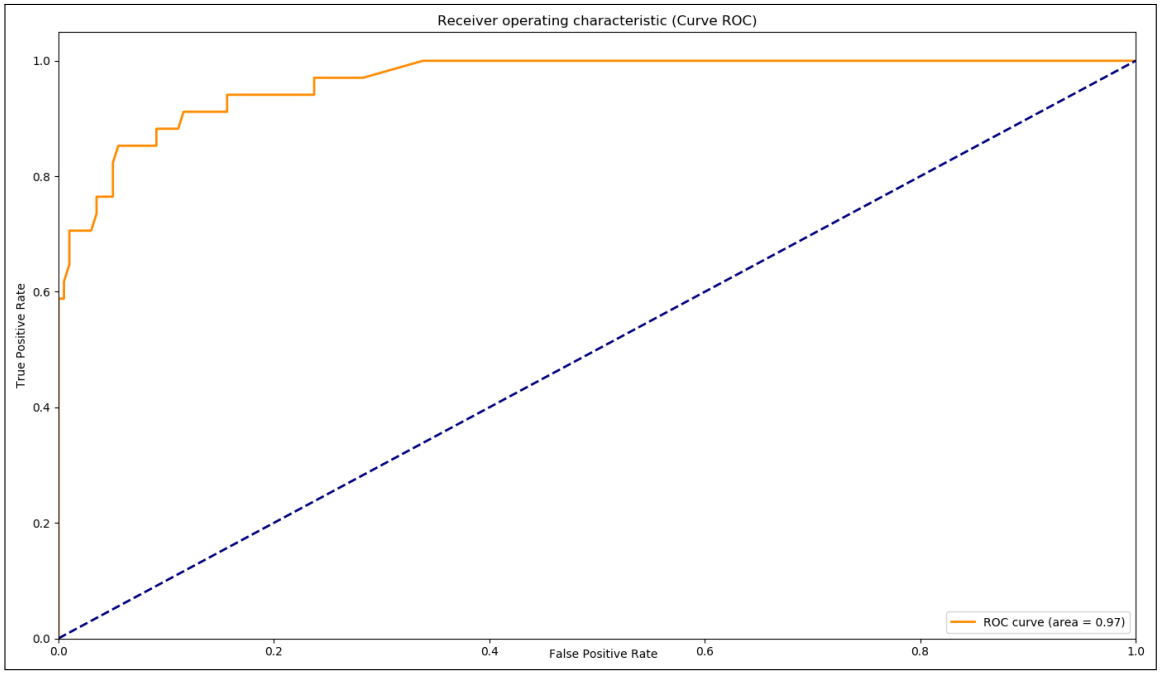
**Figura 24 – Área sob a Curva *Precision-Recall* com *Average Precision* de 74% para todos os tópicos da coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))**




Fonte: Elaborado pelo autor.

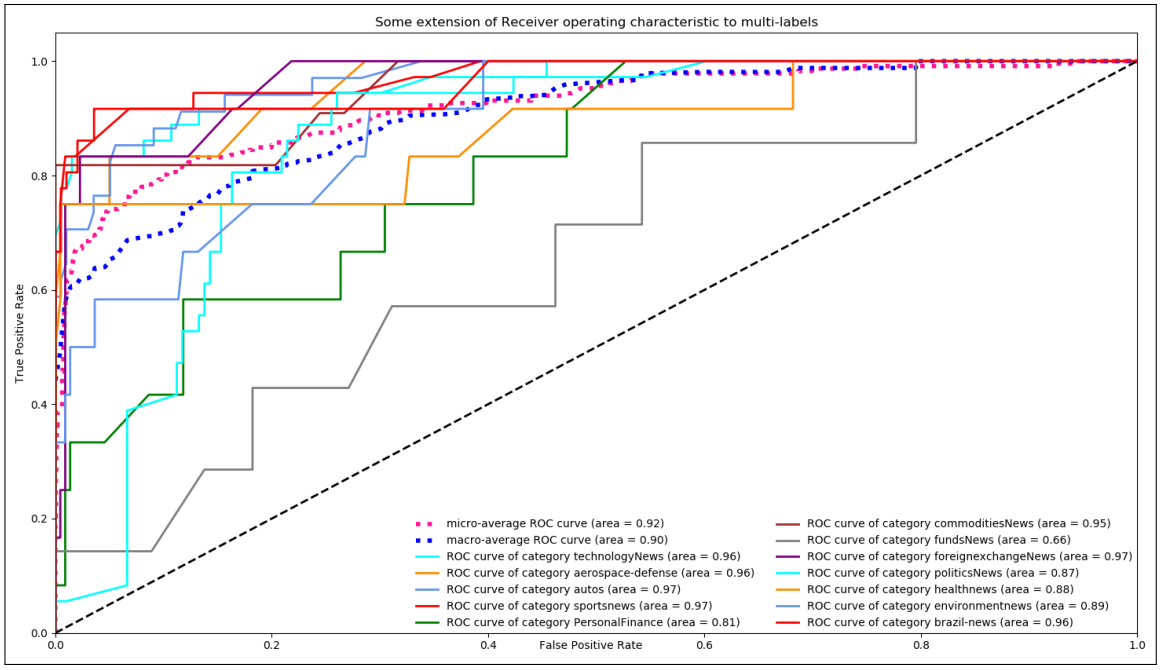
A Figura 25 mostra a curva ROC média enquanto na Figura 26 é mostrado a curva ROC detalhada. Na Figura 25, a Curva ROC média para todos os tópicos confirma o que já estava revelado na curva *precision-recall* (Figura 24). Enquanto na Figura 26, pode-se ver que a combinação tem desempenho satisfatório para a maioria dos tópicos.

**Figura 25 – Curva ROC média da coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))**




Fonte: Elaborado pelo autor.

**Figura 26 – Curva ROC detalhada por tópico para a coleção Z12News (*Online*) (combinação: SVM(RBF) +W2V (E2V-IDF))**




Fonte: Elaborado pelo autor.

### 5.3.3 Teste *Online*: Avaliação por Tópico e por Assinante

**Avaliação por Tópico** - a matriz de confusão a seguir (Tabela 22), construída com base no modelo resultante da combinação SVM(RBF)+FT (`E2V-IDF`) aplicado à coleção de documento Z5News e testado com a coleção Z5News (*Online*), tem como objetivo detalhar as métricas *Precision, Recall* e *F1-Score* por tópicos.

**Tabela 22 – Matriz de Confusão por tópicos para a coleção Z5News (*Online*) utilizando a combinação: SVM(RBF)+FT (`E2V-IDF`)**

| | *technologyNews* | *PersonalFinance* | *sportsNews* | *brazil-news* | *politicsNews* | *null* |
|---|---|---|---|---|---|---|
| *technologyNews* | **36** | 0 | 0 | 0 | 6 | 7 |
| *PersonalFinance* | 0 | **8** | 0 | 0 | 2 | 1 |
| *sportsNews* | 0 | 0 | **49** | 0 | 6 | 2 |
| *brazil-news* | 0 | 0 | 0 | **9** | 2 | 1 |
| *politicsNews* | 0 | 0 | 1 | 0 | **43** | 14 |

Fonte: Elaborado pelo autor.

A coluna *null* da Tabela 22, são notícias que não foram categorizadas em nenhum dos 5 tópicos da coleção Z5News. Isso significa que o classificador rejeitou tomar a decisão, pois a confiança desta ficou abaixo de um limiar mínimo utilizado pelo *scikit-learn* (0.5).

Com base na Matriz de Confusão da Tabela 22, a Tabela 23 mostra as métricas *Precision, Recall* e *F1-Score* por tópico.

**Tabela 23 – Métricas de Classificação baseada na matriz de confusão por tópicos para a coleção Z5News (*Online*) (combinação: SVM(RBF)+FT (`E2V-IDF`))**

| *Categorias - **Métricas*** | ***Precision*** | ***Recall*** | ***F1-Score*** |
|---|---|---|---|
| *technologyNews* | 1,00 | 0,73 | 0,85 |
| *PersonalFinance* | 1,00 | 0,73 | 0,84 |
| *sportsNews* | 0,98 | 0,86 | 0,92 |
| *brazil-news* | 1,00 | 0,75 | 0,86 |
| *politicsNews* | 0,73 | 0,74 | 0,74 |

Fonte: Elaborado pelo autor.

**Avaliação por Assinante** - Para essa avaliação, foram simuladas no ***Luppar News-Rec*** a inscrição de 40 assinantes fictícios, associando-os à tópicos aleatórios. Os resultados obtidos, utilizando as métricas *Precision* (P), *Recall* (R) e *F1-Score* (F1), encontram-se no APÊNDICE A. O teste foi realizado aplicando o modelo resultante da combinação SVM(RBF)+FT (`E2V-IDF`) à coleção de documento Z5News (*Online*).

## 5.4 DISCUSSÃO

Com vista a verificar a visão individual de cada assinante sobre a qualidade do serviço do recomendador de notícias foi obtida também uma tabela das métricas para cada um dos 40 assinantes. O desejado nesse caso é que além de médias altas elas sejam aproximadamente iguais de forma que a percepção dos assinantes sejam próximas.

Esse resultado está mostrado no Tabela A1 do Apêndice A, onde foi calculada a micro-média por assinante, para as tradicionais métricas:

- *Precision*;
- *Recall*;
- *F1-Score*.

Para a métrica *Precision* (**P**) o valor máximo foi obtido pelo assinante 3 (usr3), sendo 1.00, e o valor mínimo pelo assinante 2 (usr2), igual a 0.73; para a métrica *Recall* (**R**), o valor máximo foi de 0.86 para o assinante 14 (usr14) e o valor mínimo foi de 0.73 para o assinante 3 (usr3) e para a métrica *F1-Score* (**F1**) o valor máximo foi de 0.92 para o assinante 14 (usr14) e o mínimo foi de 0.74 para o assinante 2 (usr2). Considera-se altas essas discrepâncias de percepção entre os assinantes e este como uma oportunidade de melhoria.

As métricas utilizadas nessa pesquisa avaliam a qualidade dos resultados e não outros desempenhos gerais do SRN, como tempo de processamento, por exemplo. O ***LupparRecNews***, por possuir caráter científico e didático, exibe essas métricas na tela de resultados do Sistema Recomendador de Notícias.

A combinação do Classificador SVM (*Support Vector Machine* com *kernel* RBF) e Representações *Embedding* de documentos ponderada por IDF (`E2V-IDF`) continua obtendo os melhores resultados na fase de testes do ***LupparRecNews***, como observou-se nas tabelas 14, 15, 16, 17, 18, 19, 20 e 21. Essas foram as melhores combinação obtidas neste trabalho para todas as coleções de documentos.

Observe que para as coleções de documentos próprias Z5NewsBrasil, Z12News e Z20News (Tabelas 16, 17, 20 e 21) os melhores resultados foram alcançados utilizando a representação de documento *Embedding: Word2Vec* e não a representação *FastText* como na coleção de documentos Z5News.

Embora menos relevante, pois pode ser feito *Offline*, avaliou-se também os tempos de processamento para a geração dos modelos. Destaque para o fato da coleção Z20News ter um tempo de processamento elevado para geração do modelo, nos testes chegou a mais de 7 horas de processamento, enquanto nas demais coleções esse tempo não ultrapassa 13 minutos, como podemos ver na Tabela 24.

**Tabela 24 – Tempo em minutos para geração dos Modelos obtidos pelas combinações mais bem avaliadas e aplicadas as coleções próprias para Testes *Online***

| **Coleções** | ***RF+BoW*** | ***SVM(RBF)+FT*** (E2V-IDF) | ***SVM(RBF)+W2V*** (E2V-IDF) |
|---|---|---|---|
| Z5News | 4 | 11 | 9 |
| Z5NewsBrasil | 1 (-) | 5 | 4 |
| Z12News | 1 (-) | 13 | 13 |
| Z20News | 2 | 439 | 445 |

Fonte: Elaborado pelo autor.

# 6 CONCLUSÃO

## 6.1 CONTRIBUIÇÕES DO TRABALHO

Apresentou-se um protótipo de Sistema Recomendador de Notícias baseado no paradigma de comunicação *publish/subscriber*, o ***Luppar News-Rec***, para capturar, pré-processar e armazenar (1), classificar em tópicos (2) e distribuir (3) notícias com vista a satisfação dos assinantes. Além de uma Representação de Documento *Embedding* que foi nomeada `E2V-IDF`. O foco principal foi na fase de classificar em tópicos (2), objetivando a identificação da melhor combinação Classificador (*K-Nearest Neighbor, SVM, Decision Tree e Random Forest*) / Modelo de Representação de Documentos (BoW, TF-IDF, *Word2Vec, FastText, Word2Vec(`E2V-IDF`) e FastText(`E2V-IDF`)*) aplicados na classificação de notícias curtas em tópicos, sem a necessidade de intervenção humana, a ser utilizada pelo sistema SRN.

Para avaliar o desempenho das combinações propostas, foram conduzidos experimentos com as coleções de notícias de referência: *R8, R52, 20 Newsgroups e Reuters RCV1*, além das coleções de documentos próprias: Z5News, Z5NewsBrasil, Z12News e Z20News. De maneira geral, os melhores resultados foram alcançados pela combinação *SVM* (*kernel*: RBF) com *Embeddings* utilizando a abordagem `E2V-IDF`.

Os resultados aqui obtidos são comparáveis, ou melhores em alguns casos, aqueles obtidos em publicações recentes que utilizaram as mesmas coleções de documentos. Por exemplo, em (PAWAR; GAWANDE, 2012) os classificadores *SVM* (*kernel*: RBF), KNN e Árvore de Decisão usando a coleção de documento *Reuters-21578 (R52)*, obteve os seguintes resultados respectivamente: 86.50%, 78.88% e 87.90% de *F1-Score (micro-média)*, enquanto neste trabalho obteve-se 91.07%, 90.70% e 86.15%, respectivamente. Em (SHANKAR; KARYPIS, 2000), os resultados mais expressivos obtidos para a coleção de documento *Reuters-21578 (R52)* foi usando SVM(RBF), onde a medição *Precision/Recall* chegou a 86.62% para todas as classes, enquanto a abordagem *FWA* (proposta por Eles) obteve 81.40%, já nossa abordagem proposta (`E2V-IDF`) chegou a 91.07% de *Precision/Recall*. Isso mostra que essa implementação é competitiva com o estado da arte.

A avaliação das combinações mostrou sua eficácia tanto utilizando notícias em Inglês quanto em Português. As combinações melhor avaliadas foram utilizando o Classificador SVM com *kernel*: RBF em conjunto com Representações de Documentos *Embedding (FastText ou Word2Vec)* utilizando a abordagem ponderada por IDF (`E2V-IDF`).

Em resumo, as contribuições deste trabalho foram a proposição de uma estrutura e o desenvolvimento de um SRN, o ***Luppar News-Rec***, de forma estruturada e exequível cujo código fonte está disponibilizado em seu sítio (http://luppar.com/recommender); e uma abordagem *embedding* - `E2V-IDF` implementada no ***Luppar News-Rec***. A exaustiva avaliação de combinações de representação de documentos com classificadores mostrada em diversas tabelas ao longo do trabalho pode também ser utilizada por outros projetistas para guiar a construção de SRNs.

## 6.2 LIMITAÇÕES

As principais limitações identificadas durante o desenvolvimento do trabalho foram: no Sistema de captura, pré-processamento e armazenamento, que baixam notícias de um número limitado de fontes de notícias (EN | PT-BR); no Subsistema de distribuição que não utiliza uma abordagem para recomendação de notícias com base em similaridade de perfis (filtragem colaborativa) por exemplo. Deve-se notar que os testes do sistema também foram limitados já que as coleções de documentos próprias são monorrótulo. Essa limitação deveu-se a que foi utilizada a rotulação de origem das fontes de notícias e essas só forneciam monorótulo. A solução seria a rotulação manual pelo autor, o que tornou-se inviável.

## 6.3 TRABALHOS FUTUROS

A evolução em andamento deste trabalho está sendo na implementação do Sistema Recomendador de Notícias com Aprendizagem de Perfil (Filtragem Colaborativa) e em melhorias na representação de documento `E2V-IDF`, para com isso tirar melhor proveito da densidade semântica de *embeddings* e tentar obter resultados ainda mais expressivos. Outro ponto em andamento é a multirrotulação automática de notícias. Além disso, como mostrado na Tabela A1, essa implementação ainda pode gerar discrepâncias de percepção na qualidade de serviço entre diferentes assinantes. Esta é uma deficiência do subsistema de classificação. Então, pretende-se também ainda investigar melhorias neste subsistema.

# REFERÊNCIAS

ADHIANTO, L.; BANERJEE, S.; FAGAN, M.; KRENTEL, M.; MARIN, G.; MELLOR-CRUMMEY, J.; TALLENT, N. R. Hpctoolkit: tools for performance analysis of optimized parallel programs. **Concurrency and Computation: Practice and Experience**, Wiley Online Library, v. 22, n. 6, p. 685–701, 2010.

ADOMAVICIUS, G.; MOBASHER, B.; RICCI, F.; TUZHILIN, A. Context-aware recommender systems. **AI Magazine**, Springer, v. 32, n. 3, p. 217–253, 2011.

AHN, J.; BRUSILOVSKY, P.; GRADY, J.; HE, D.; SYN, S. Y. Open user profiles for adaptive news systems: help or harm? **Proceedings of the 16th international conference on World Wide Web**, ACM, p. 11–20, 2007.

ANAND, S. S.; MOBASHER, B. Contextual recommendation. **Workshop on Web Mining**, Springer, p. 142–160, 2006.

ARTIFICIAL INTELLIGENCE IN MOTION. **Introdução a árvores de decisão para classificação e mineração de dados**. 2009. Disponível em: <http://aimotion.blogspot.com.br/2009/04/artigo-introducao-arvores-de-decisao.html>. Acesso em: 25 set. 2019.

BAEZA-YATES, R.; RIBEIRO-NETO, B. **Recuperação de informação**: conceitos e tecnologia das máquinas de busca. 2. ed. Porto Alegre: Bookman Editora, 2013. ISBN 978-85-8260-049-8.

BARUAH, G.; KOLLA, M. Klick labs at cl-scisumm 2018. **Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries**, CEUR Workshop Proceedings, p. 134–141, 2018.

BOJANOWSKI, P.; GRAVE, E.; JOULIN, A.; MIKOLOV, T. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. **Proceedings of the fifth annual workshop on Computational learning theory**, ACM, p. 144–152, 1992.

BURKE, R. Hybrid web recommender systems. **The adaptive web**, Springer, p. 377–408, 2007.

DAS, A. S.; DATAR, M.; GARG, A.; RAJARAM, S. Google news personalization: scalable online collaborative filtering. **International conference on World Wide Web**, ACM, v. 16, p. 271–280, 2007.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern classification and scene analysis**. 2. ed. [S.l.]: Wiley Interscience, 1995.

FONSECA, T. L. *et al.* **Algoritmo genético com regressão**. Dissertação (Mestrado em Modelagem Computacional) — Universidade Federal de Juiz de Fora, 2017.

GARCIN, F.; ZHOU, K.; FALTINGS, B.; SCHICKEL, V. Personalized news recommendation based on collaborative filtering. **International Conferences on Web Intelligence and Intelligent Agent Technology**, IEEE, v. 1, p. 437–441, 2012.

GULLA, J. A.; ALMEROTH, K. C.; TAVAKOLIFARD, M.; HOPFGARTNER, F. **Proceedings of the 2013 International News Recommender Systems Workshop and Challenge**. [S.l.]: ACM, 2013.

HIDASI, B.; KARATZOGLOU, A.; BALTRUNAS, L.; TIKK, D. Session-based recommendations with recurrent neural networks. **Computing Research Repository**, 2015.

HSU, C. W.; LIN, C. J. A comparison of methods for multiclass support vector machines. **IEEE transactions on Neural Networks**, IEEE, v. 13, n. 2, p. 415–425, 2002.

HU, G. Personalized neural embeddings for collaborative filtering with text. **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**, Association for Computational Linguistics, v. 1, p. 2082–2088, 2019.

IJNTEMA, W.; GOOSSEN, F.; FRASINCAR, F.; HOGENBOOM, F. Ontology-based news recommendation. **Proceedings of the 2010 EDBT/ICDT Workshops**, ACM, p. 16, 2010.

KALMEGH, S. Analysis of weka data mining algorithm reptree, simple cart and randomtree for classification of indian news. **International Journal of Innovative Science, Engineering & Technology**, v. 2, n. 2, p. 438–446, 2015.

KARATZOGLOU, A.; HIDASI, B. Deep learning for recommender systems. **Proceedings of the eleventh ACM conference on recommender systems**, ACM, p. 396–397, 2017.

KAUR, G.; KAUR, E. P. Novel approach to text classification by svm-rbf kernel and linear svc. **International Journal of Advance Research, Ideas and Innovation in Technology**, v. 3, n. 3, 2017.

KHATTAR, D.; KUMAR, V.; GUPTA, M.; VARMA, V. Neural content-collaborative filtering for news recommendation. **NewsIR@ ECIR**, v. 2079, p. 45–50, 2018.

KIM, H. K.; KIM, H.; CHO, S. Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. **Neurocomputing**, Elsevier, v. 266, p. 336–352, 2017.

KIM, J.; KIM, K. Integrated model of economic generation system expansion plan for the stable operation of a power plant and the response of future electricity power demand. **Journal Sustainability**, Multidisciplinary Digital Publishing Institute, v. 10, n. 7, p. 2417, 2018.

KOMPAN, M.; BIELIKOVÁ, M. Content-based news recommendation. **International conference on electronic commerce and web technologies**, Springer, p. 61–72, 2010.

KOSALA, R.; BLOCKEEL, H. Web mining research: a survey. **ACM Sigkdd Explorations Newsletter**, ACM, v. 2, n. 1, p. 1–15, 2000.

LIU, J.; DOLAN, P.; PEDERSEN, E. R. Personalized news recommendation based on click behavior. **International conference on Intelligent user interfaces**, ACM, v. 15, p. 31–40, 2010.

LOMMATZSCH, A.; ALBAYRAK, S. Real-time recommendations for user-item streams. **Annual ACM Symposium on Applied Computing**, ACM, v. 30, p. 1039–1046, 2015.

LORENA, A. C.; CARVALHO, A. C. de. Uma introdução às support vector machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43–67, 2007.

LUCCA, J. D.; NUNES, M. G. V. Lematização versus stemming. **Interinstitutional Center for Computational Linguistics**, USP, 2002.

MA, H.; LIU, X.; SHEN, Z. User fatigue in online news recommendation. **International Conference on World Wide Web**, ACM, v. 25, p. 1363–1372, 2016.

MARTINČIĆ-IPŠIĆ, S.; MILIČIĆ, T.; TODOROVSKI, L. The influence of feature representation of text on the performance of document classification. **Applied Sciences**, IACSIT Press, p. 743, 2019.

MEDIUM. **O que é o processamento de linguagem natural?** 2017. Disponível em: <https://medium.com/botsbrasil/o-que-é-o-processamento-de-linguagem-natural-49ece9371cff>. Acesso em: 25 set. 2019.

MIKOLOV, T.; CHEN, K.; CORRADO, G. S.; DEAN, J. Efficient estimation of word representations in vector space. **Computing Research Repository**, 2013.

MOREIRA, G. S. P.; FERREIRA, F.; CUNHA, A. M. News session-based recommendations using deep neural networks. **Workshop on Deep Learning for Recommender Systems**, ACM, v. 3, p. 15–23, 2018.

MUSTO, C.; SEMERARO, G.; GEMMIS, M.; LOPS, P. Learning word embeddings from wikipedia for content-based recommender systems. **European Conference on Information Retrieval**, Springer, p. 729–734, 2016.

NANDI, R. N.; ZAMAN, M. A.; MUNTASIR, T. A.; SUMIT, S. H.; SOUROV, T.; RAHMAN, M. J.-U. Bangla news recommendation using doc2vec. **International Conference on Bangla Speech and Language Processing**, IEEE, p. 1–5, 2018.

NORVIG, P.; RUSSELL, S. **Inteligência Artificial**. [S.l.]: Elsevier Brasil, 2014.

OH, K. J.; LEE, W. J.; LIM, C. G.; CHOI, H. J. Personalized news recommendation using classified keywords to capture user preference. **International Conference on Advanced Communication Technology**, IEEE, v. 16, p. 1283–1287, 2014.

ÖZGÖBEK, Ö.; GULLA, J. A.; ERDUR, R. C. A survey on challenges and methods in news recommendation. **International Conference on Web Information Systems and Technologies**, Springer, v. 10, p. 278–285, 2014.

PASCHOALIN, M. A.; SPADOTO, N. T. **Gramática**: teoria e exercícios. 1. ed. [S.l.]: FTD Editora, 1996.

PASCOAL, T.; BRITO, D. M.; ANDRADE, L.; RÊGO, T. G. Evasao de estudantes universitários: diagnóstico a partir de dados acadêmicos e socieconômicos. **Simpósio Brasileiro de Informática na Educação**, v. 27, n. 1, p. 926, 2016.

PAWAR, P. Y.; GAWANDE, S. A comparative study on different types of approaches to text categorization. **International Journal of Machine Learning and Computing**, IACSIT Press, v. 2, n. 4, p. 423, 2012.

PAZZANI, M. J. A framework for collaborative, content-based and demographic filtering. **Artificial intelligence review**, Springer, v. 13, n. 5-6, p. 393–408, 1999.

PISSANETZKY, S. **Sparse Matrix Technology**. [S.l.]: Academic Press, 1984.

QUADRANA, M.; KARATZOGLOU, A.; HIDASI, B.; CREMONESI, P. Personalizing session-based recommendations with hierarchical recurrent neural networks. **Proceedings of the Eleventh ACM Conference on Recommender Systems**, ACM, p. 130–137, 2017.

REI, M.; CUMMINS, R. Sentence similarity measures for fine-grained estimation of topical relevance in learner essays. **Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications**, Association for Computational Linguistics, p. 283–288, 2016.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence**. 3. ed. [S.l.]: Pearson Education, 2016.

SAMARINAS, C.; ZAFEIRIOU, S. Personalized high quality news recommendations using word embeddings and text classification models. **EasyChair Events**, EasyChair, 2019.

SANTANA, B. S. **Extração e aplicação de indicadores no processo de recomendação de recursos urbanos utilizando dados estruturados e não-estruturados**. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul, 2019.

SANTOS, C. D.; GATTI, M. Deep convolutional neural networks for sentiment analysis of short texts. **International Conference on Computational Linguistics**, v. 25, p. 69–78, 2014.

SCHOLKOPF, B.; SMOLA, A. J. **Learning with kernels**: support vector machines, regularization, optimization, and beyond. [S.l.]: MIT press, 2001.

SCHÜTZE, H.; MANNING, C. D.; RAGHAVAN, P. **Introduction to information retrieval**. [S.l.]: Cambridge University Press, 2008.

SEBASTIANI, F. Machine learning in automated text categorization. **ACM computing surveys**, ACM, v. 34, n. 1, p. 1–47, 2002.

SHAHI, T. B.; PANT, A. K. Nepali news classification using naïve bayes, support vector machines and neural networks. **International Conference on Communication information and Computing Technology**, IEEE, p. 1–5, 2018.

SHANKAR, S.; KARYPIS, G. A feature weight adjustment algorithm for document categorization. **KDD-2000 Workshop on Text Mining**, Carbegie Mellon, 2000.

SHARDANAND, U.; MAES, P. Social information filtering: algorithms for automating "word of mouth". **CHI '95 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**, ACM, p. 210–217, 1995.

SILVA, F. T. **Luppar**: um sistema de recuperação de informação para coleções fechadas de documentos. Dissertação (Mestrado Acadêmico em Ciência da Computação) — Universidade Estadual do Ceará, 2018.

SILVA, R. G. N. **Sistema de Recomendação baseado em conteúdo textual**. Dissertação (Mestrado Multiinstitucional em Ciência da Computação) — Universidade Federal da Bahia, 2014.

SOKOLOVA, M.; LAPALME, G. A systematic analysis of performance measures for classification tasks. **Information processing & management**, Elsevier, v. 45, n. 4, p. 427–437, 2009.

SOUSA, S. **Estudo de modelos de word embedding**. 2016. 53 p. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) — Universidade Tecnológica Federal do Paraná, 2016.

SOUZA, A.; MAIA, J. E. B. Agente inteligente para classificação de notícias por assunto. **Computer on the Beach**, p. 714–723, 2019.

SOUZA, A. H. *et al.* **Seleção de atributos relevantes - aplicando técnicas na base de dados do herbário virtual da flora e dos fungos**. Dissertação (Mestrado em Ciência e Tecnologia para Recursos Amazônicos) — Universidade Federal do Amazonas, 2017.

SU, X.; KHOSHGOFTAAR, T. M. A survey of collaborative filtering techniques. **Advances in artificial intelligence**, Hindawi, v. 2009, 2009.

TELOKEN, A. *et al.* Estudo comparativo entre os algoritmos de mineração de dados random forest e j48 na tomada de decisão. **Simpósio de Pesquisa e Desenvolvimento em Computação**, v. 2, n. 1, 2016.

THAOROIJAM, K. A study on document classification using machine learning techniques. **International Journal of Computer Science Issues**, International Journal of Computer Science Issues, v. 11, n. 2, p. 217, 2014.

UDACITY. **Decision Trees with scikit-learn**. 2016. Disponível em: <http://napitupulu-jon.appspot.com/posts/decision-tree-ud.html>. Acesso em: 19 jul. 2019.

WANG, Y.; WANG, X.-J. A new approach to feature selection in text classification. **International Conference of Machine Learning and Cybernetics**, IEEE, v. 6, p. 3814–3819, 2005.

ZHANG, Y.; JIN, R.; ZHOU, Z.-H. Understanding bag-of-words model: a statistical framework. **International Journal of Machine Learning and Cybernetics**, Springer, v. 1, n. 1-4, p. 43–52, 2010.

ZHENG, L.; LI, L.; HONG, W.; LI, T. Penetrate: Personalized news recommendation using ensemble hierarchical clustering. **Expert Systems with Applications**, Elsevier, v. 40, n. 6, p. 2127–2136, 2013.

ZHU, X. Semi-supervised learning literature survey. **Computer Science - University of Wisconsin-Madison**, v. 1, 2005.

# APÊNDICES

# APÊNDICE A – SIMULAÇÃO DE PERFIS DE ASSINANTES

**Tabela A1 – Simulação de Perfis de Assinantes (SVM(RBF)+FT (`E2V-IDF`))**

| Coleção Z5News (*Online*) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| E-mail | **sports** | **politics** | **technology** | **PersonalFinance** | **brazil** | **P** | **R** | **F1** |
| usr1@nrec.com | X | | X | | | 0,99 | 0,80 | 0,89 |
| usr2@nrec.com | | X | | | | 0,73 | 0,74 | 0,74 |
| usr3@nrec.com | | | X | | | **1,00** | 0,73 | 0,85 |
| usr4@nrec.com | X | | | | X | 0,99 | 0,81 | 0,89 |
| usr5@nrec.com | | X | X | | | 0,87 | 0,74 | 0,80 |
| usr6@nrec.com | | | X | | X | **1,00** | 0,74 | 0,86 |
| usr7@nrec.com | | X | | | X | 0,87 | 0,75 | 0,80 |
| usr8@nrec.com | X | X | | | | 0,86 | 0,80 | 0,83 |
| usr9@nrec.com | X | X | X | X | X | 0,94 | 0,76 | 0,84 |
| usr10@nrec.com | X | | | X | | 0,99 | 0,80 | 0,88 |
| usr11@nrec.com | | | X | | X | **1,00** | 0,74 | 0,86 |
| usr12@nrec.com | | X | | X | X | 0,91 | 0,74 | 0,81 |
| usr13@nrec.com | | | X | | | **1,00** | 0,73 | 0,85 |
| usr14@nrec.com | X | | | | | 0,98 | **0,86** | **0,92** |
| usr15@nrec.com | X | | | X | X | 0,99 | 0,78 | 0,87 |
| usr16@nrec.com | | X | X | | | 0,87 | 0,74 | 0,80 |
| usr17@nrec.com | | X | | | X | 0,87 | 0,75 | 0,80 |
| usr18@nrec.com | | | X | | X | **1,00** | 0,74 | 0,86 |
| usr19@nrec.com | X | X | X | | | 0,90 | 0,78 | 0,84 |
| usr20@nrec.com | | | X | X | X | **1,00** | 0,74 | 0,85 |
| usr21@nrec.com | | X | X | | | 0,87 | 0,74 | 0,80 |
| usr22@nrec.com | | | X | | | **1,00** | 0,73 | 0,85 |
| usr23@nrec.com | | | | X | | **1,00** | 0,73 | 0,84 |
| usr24@nrec.com | | X | | X | | 0,87 | 0,74 | 0,79 |
| usr25@nrec.com | X | | | X | | 0,99 | 0,80 | 0,88 |
| usr26@nrec.com | | X | X | | | 0,87 | 0,74 | 0,80 |
| usr27@nrec.com | X | | | X | | 0,99 | 0,80 | 0,88 |
| usr28@nrec.com | X | X | X | X | X | 0,94 | 0,76 | 0,84 |
| usr29@nrec.com | X | | X | | X | 0,99 | 0,78 | 0,88 |
| usr30@nrec.com | X | | | X | X | **1,00** | 0,74 | 0,85 |
| usr31@nrec.com | | | X | X | | **1,00** | 0,73 | 0,85 |
| usr32@nrec.com | X | X | | | X | 0,90 | 0,78 | 0,84 |
| usr33@nrec.com | | | X | X | | **1,00** | 0,73 | 0,85 |
| usr34@nrec.com | | | | | X | **1,00** | 0,75 | 0,86 |
| usr35@nrec.com | | | X | X | X | **1,00** | 0,74 | 0,85 |
| usr36@nrec.com | X | X | | | | 0,86 | 0,80 | 0,83 |
| usr37@nrec.com | | X | X | | | 0,87 | 0,74 | 0,80 |
| usr38@nrec.com | | | | | X | **1,00** | 0,75 | 0,86 |
| usr39@nrec.com | X | X | X | | | 0,90 | 0,78 | 0,84 |
| usr40@nrec.com | | | X | X | X | **1,00** | 0,74 | 0,85 |

Fonte: Elaborado pelo autor.

# APÊNDICE B – COLEÇÕES PRÓPRIAS - DETALHAMENTO POR TÓPICOS

**Tabela B1 – Número de notícias para: Treinamento (Teste *Offline* | Teste *Online*)**

| Detalhamento por tópicos das coleções próprias | | | | |
|---|---|---|---|---|
| **Tópicos** | **Z5 (*Off* \| *On* )** | **Z5BR (*Off* \| *On* )** | **Z12 (*Off* \| *On* )** | **Z20 (*Off* \| *On* )** |
| technologyNews | 8717 (150 \| 49) | | 3770 (130 \| 36) | 6931 (130 \| 36) |
| sportsnews | 8539 (146 \| 57) | | 2309 (131 \| 36) | 6758 (131 \| 36) |
| PersonalFinance | 8335 (158 \| 11) | | 3446 (102 \| 12) | 6564 (102 \| 12) |
| politicsNews | 8756 (148 \| 58) | | 4125 (124 \| 36) | 6322 (124 \| 36) |
| brazil-news | 8185 (167 \| 12) | | 3917 (108 \| 12) | 6705 (108 \| 12) |
| aerospace-defense | | | 3730 (107 \| 12) | 6902 (107 \| 12) |
| autos | | | 3926 (130 \| 34) | 6892 (130 \| 34) |
| commoditiesNews | | | 3973 (106 \| 11) | 6861 (106 \| 11) |
| fundsNews | | | 3969 (098 \| 07) | 6436 (098 \| 07) |
| foreignexchangeNews | | | 2051 (108 \| 12) | 3455 (108 \| 12) |
| healthnews | | | 3213 (108 \| 12) | 6892 (108 \| 12) |
| environmentnews | | | 3420 (108 \| 12) | 6886 (108 \| 12) |
| innovationNews | | | | 6953 (108 \| 12) |
| sustainable_business | | | | 0992 (108 \| 12) |
| businessNews | | | | 6866 (131 \| 37) |
| mcbreakingviews | | | | 3030 (108 \| 12) |
| banks | | | | 4554 (106 \| 12) |
| marketsNews | | | | 5876 (091 \| 10) |
| wealth-retirement | | | | 4244 (101 \| 12) |
| worldNews | | | | 6908 (133 \| 37) |
| esporteNews | | 3897 (135 \| 42) | | |
| politicaNews | | 5153 (106 \| 34) | | |
| tecnologiaNews | | 4516 (142 \| 44) | | |
| financaPessoal | | 4057 (139 \| 40) | | |
| educacaonews | | 4554 (134 \| 29) | | |
| **Número de Notícias** | **42532 (769 \| 187)** | **22177 (656 \| 189)** | **41849 (1360 \| 232)** | **117027 (2246 \| 376)** |

Fonte: Elaborado pelo autor.

# APÊNDICE C – TESTE INTER-BASES

**Teste Inter-bases** - Denominamos de Teste Inter-Bases a questão de Treinar o Modelo, de *Machine Learning*, com uma coleção de documentos de uma determinada fonte de dados (no caso, de um site de notícias) e Testar (predizer) com uma outra coleção de documentos de outra fonte de dados (no caso, outros sites de notícias, podendo ser incluído notícias do site escolhido para treinamento, mas que sejam notícias ainda não vistas).

O intuito desse teste é que não seja preciso treinar um modelo e testar (predizer) sempre utilizando uma mesma fonte de dados, mas sim, poder treinar um modelo com uma determinada fonte de dados e os testes poderem ser realizados utilizando qualquer outra fonte de dados, independente do estilo de escrita, padrões gramaticais, etc.

Coleções de documentos utilizadas (maiores detalhes, na Tabela 2):

- **Z5News** (Treinamento - Colhidas do site de notícias: Reuters);
- **Z5News *Online*** - (Teste - Colhidas dos sites de notícias: Reuters e Inshorts);
- **Z5NewsBrasil** (Treinamento - Colhidas do site de notícias: G1 Notícias);
- **Z5NewsBrasil *Online*** - (Teste - Colhidas do site de notícias: UOL Notícias);
- **Z12News** (Treinamento - Colhidas do site de notícias: Reuters);
- **Z12News *Online*** - (Teste - Colhidas dos sites de notícias: Reuters e Inshorts).

**Treinamento x Teste** - Foram realizados comparativos de desempenho (*F1-Score*) para cada uma das coleções de documentos. A primeira avaliação, Treinando e Testando com dados da mesma coleção (dados de teste não vistos no treinamento), Tabela C1.

**Tabela C1 – Treinando e Testando com dados da mesma coleção (dados de teste não vistos no treinamento)**

| Combinação | Z5News | Z5NewsBrasil | Z12News |
|---|---|---|---|
| ***RF+BoW*** | 0,7818 | 0,7651 | 0,5671 |
| ***SVM(RBF)+FT*** (`E2V-IDF`) | 0,7846 | 0,7865 | 0,5828 |
| ***SVM(RBF)+W2V*** (`E2V-IDF`) | 0,7838 | 0,7920 | 0,5954 |

Fonte: Elaborado pelo autor.

E a segunda avaliação, Treinando com dados de uma coleção e Testando com dados de outra coleção (também foi incluso nos testes dados da mesma coleção de treinamento, dados esses não vistos no treinamento), Tabela C2.

**Tabela C2 – Treinando com dados de uma coleção e Testando com dados de outra coleção (também foi incluso, nos testes, dados da mesma coleção de treinamento, dados esses não vistos no treinamento)**

| Combinação | Z5News | Z5NewsBrasil | Z12News |
|---|---|---|---|
| *RF+BoW* | 0,7273 | 0,6745 | 0,4699 |
| ***SVM(RBF)+FT*** (E2V-IDF) | **0,8309** | 0,7450 | **0,6595** |
| ***SVM(RBF)+W2V*** (E2V-IDF) | **0,8047** | 0,7335 | **0,6614** |

Fonte: Elaborado pelo autor.

Observamos na Tabela C2, que para as combinações envolvendo representações *embeddings* de documentos (destacadas em **negrito**), as coleções **Z5News** e **Z12News** tiveram ganhos de performance, quando comparada com a Tabela C1, mesmo usando para testes dados de outras fontes (que podem possuir padrões de escrita diferentes em relação a fonte de dados de treinamento). A **Z5NewsBrasil** não obteve ganho de performance, essa coleção foi testada apenas com dados de outra fonte, não foi incluso dados da fonte de treinamento, como ocorreu com a **Z5News** e **Z12News**.