ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

HOLISSON SOARES DA CUNHA

SENTIMENTSTREAM: UM COMITÊ DE CLASSIFICADORES ADAPTATIVO PARA ANÁLISE DE SENTIMENTO DE TWEETS

Porto Alegre
2016

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

SENTIMENTSTREAM: UM COMITÊ DE CLASSIFICADORES ADAPTATIVO PARA ANÁLISE DE SENTIMENTO DE TWEETS

HOLISSON SOARES DA CUNHA

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Ciência da Computação na Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: Prof. Dr. Duncan Dubugras Alcoba Ruiz

Porto Alegre
2016

Ficha Catalográfica

C972s Cunha, Holisson Soares da

SENTIMENTSTREAM Um Comitê de Classificadores Adaptativo para Análise de Sentimento de Tweets / Holisson Soares da Cunha. – 2016.
84 p.
Dissertação (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientador: Prof. Dr. Duncan Dubugras Alcoba Ruiz.

1. Análise de Sentimento. 2. Mineração em Fluxo Contínuo de Dados. 3. Aprendizado de Máquina. 4. Concept Drift. 5. Feature Drift. I. Ruiz, Duncan Dubugras Alcoba. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS
com os dados fornecidos pelo(a) autor(a).
Bibliotecária responsável: Loiva Duarte Novak CRB-10/2079

HOLISSON SOARES DA CUNHA

SENTIMENTSTREAM: UM COMITÊ DE CLASSIFICADORES ADAPTATIVO PARA ANÁLISE DE SENTIMENTO DE TWEETS

Dissertação apresentada como requisito parcial para obtenção do grau de Mestre em Ciência da Computação do Programa de Pós-Graduação em Ciência da Computação, Escola Politécnica da Pontifícia Universidade Católica do Rio Grande do Sul.

Aprovado(a) em 30 de Abril de 2016.

BANCA EXAMINADORA:

Profa. Dra. Karin Becker (UFRGS)

Prof. Dr. Rodrigo Coelho Barros (PPGCC/PUCRS)

Prof. Dr. Duncan Dubugras Alcoba Ruiz (PPGCC/PUCRS - Orientador)

AGRADECIMENTOS

Ao meu orientador, professor Dr. Duncan Dubugras Alcoba Ruiz, pelo suporte, aprendizado e paciência em todas as situações do mestrado. Sou extremamente grato pela oportunidade e pela experiência.

Ao professor Dr. Rodrigo Coelho Barros, pelo aprendizado e cooperação nesse trabalho.

A minha família, especialmente a minha esposa, Graziela, por toda paciência e incentivo.

Aos colegas do laboratório, especialmente a Renata De Paris e Christian Quevedo, que me apoiaram em vários momentos da pesquisa, compartilhando seu tempo e conhecimento.

Ao PPGCC e a FAPERGS, que tornaram possível a realização desta pesquisa, através do auxílio financeiro.

SENTIMENTSTREAM: UM COMITÊ DE CLASSIFICADORES ADAPTATIVO PARA ANÁLISE DE SENTIMENTO DE TWEETS

RESUMO

Diariamente, milhões de usuários utilizam o Twitter para compartilhar mensagens, fornecendo um enorme volume de conteúdo opinativo sobre diversos tópicos de interesse da sociedade. Além da quantidade de mensagens, o Twitter caracteriza-se como uma rede social de Fluxo Contínuo de Dados, que gera novas mensagens em tempo real, em alta velocidade e com distribuição não estacionária. Devido a essas características, pesquisas recentes em Análise de Sentimento têm explorado o Twitter em tarefas de classificação *online*, considerando restrições de tempo, memória e a necessidade de adaptação às mudanças que podem ocorrer na distribuição dos dados. Chamado de *Concept Drift*, esse fenômeno ocorre em decorrência de potenciais mudanças na distribuição que gera novos dados dentro do fluxo, afetando diretamente a capacidade de generalização do algoritmo. Além disso, a Análise de Sentimento introduz um tipo especial de mudança, chamada de *Feature Drift*. Trata-se de um problema onde novos atributos relevantes são encontrados ao longo do fluxo e atributos conhecidos se tornam irrelevantes, o que sugere o uso de um espaço dimensional dinâmico. Com base nesses desafios, neste trabalho é proposto SENTIMENTSTREAM, um comitê de classificadores dinâmico, baseado em lotes de dados, e que incrementalmente processa e avalia novas instâncias ao longo do fluxo. Especializado na classificação de *tweets*, SENTIMENTSTREAM é composto por dois componentes principais: (i) Um detector de *concept drift*, capaz de detectar e reagir de forma eficiente a mudanças abruptas na distribuição dos dados e, (ii) um detector de *feature drift*, que utiliza uma estratégia automática para monitorar e identificar potenciais mudanças no espaço de atributos. Experimentos com dados reais do Twitter indicam que SENTIMENTSTREAM apresenta resultados efetivos, sendo eficaz no processo de classificação de *tweets* e no tratamento de mudanças abruptas na distribuição dos dados.

Palavras-Chave: Análise de Sentimento, Mineração em Fluxo Contínuo de Dados, Aprendizado de Máquina, Espaço Dinâmico de Atributos, *Concept Drift*, *Feature Drift*, Comitê de Classificadores.

SENTIMENTSTREAM: AN ADAPTATIVE ENSEMBLE CLASSIFIERS FOR SENTIMENT ANALYSIS ON TWEETS

ABSTRACT

Daily, millions of users use Twitter to share messages, providing a huge amount of opinionated content on various topics of interest to society. In addition to the volume of messages, Twitter is characterized as a social network in data streaming, that generates new messages in real-time at high speed and with a nonstationary distribution. Because of these characteristics, recent research in Sentiment Analysis has explored Twitter as an online classification task, considering constraints of time, memory, and the need to adapt to changes that may occur in the data distribution. Called concept drift, this phenomenon occurs due to potential changes in the distribution that generates new data within the stream, directly affecting the algorithm's ability to generalize. Furthermore, the Sentiment Analysis introduces a special kind of challenge, called feature drift. In this case, new relevant attributes are found along the stream and known attributes may become irrelevant, which suggests the use of dynamic feature space. Based on these challenges, this work proposes SENTIMENTSTREAM, a dynamic ensemble classifier, which incrementally processes and analyses new instances along the stream. Specialized to process Twitter data, SENTIMENTSTREAM is composed of two main components: (i) A concept drift detector, able to detect and react efficiently to abrupt changes in the data distribution, and (ii) a feature drift detector, which uses an automatic strategy to monitor and identify potential changes in the attributes space. Experimentation with real data of Twitter indicates that Twitter SENTIMENTSTREAM presents effective results, being effective for tweets classification and treatment of potential changes in the data distribution.

Keywords: Sentiment Analysis, Data Stream Mining, Machine Learning, Dynamic Feature Space, Concept Drift, Feature Drift, Ensemble Classifiers.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE SIGLAS

AM – Aprendizado de Máquina

AS – Análise de Sentimento

BOW – *Bag-of-Words*

DCD – Detector de *Concept Drift*

DFD – Detector de *Feature Drift*

DWM – Dynamic Weighted Majority

FCD – Fluxo Contínuo de Dados

HT – Hoeffding Tree

IDN – ÍnDice de Novidade

MNB – Multinomial Naïve Bayes

MOA – Massive Online Analysis

PLN – Processamento de Linguagem Natural

POS – *Part-of-Speech*

SEA – *Streaming Ensemble Algorithm*

SVM – *Support Vector Machine*

TF – *Term-Frequency*

TF-IDF – *Term-Frequency Inverse Document Frequency*

VFDT – Very Fast Decision Tree

WMA – Weighted Majority Algorithm

SUMÁRIO

1. INTRODUÇÃO

Nos últimos anos as redes sociais se tornaram poderosas fontes de compartilhamento de mensagens e conteúdo opinativo. Segundo dados fornecidos pelo Twitter, em 2016, o número de usuários alcançou 300 milhões, chegando também a marca de 1 bilhão de visitas mensais no site [63]. Além de ser uma ferramenta de comunicação, usuários fazem das redes sociais um ambiente de discussão, reflexão e compartilhamento de opiniões sobre tópicos emergentes, experiências de consumo e diferentes assuntos que fazem parte de seus cotidianos. No Twitter, as mensagens compartilhadas (chamadas de *tweets*), podem expressar o sentimento do público *online* sobre candidatos em campanhas eleitorais [64], analisar o interesse de consumo dos usuários [28], reconhecer o sentimento do público em grandes eventos [60] ou apoiar na previsão das flutuações em cotações financeiras [12]. Devido a essas possibilidades e a ascensão das redes sociais, aplicações para Análise de Sentimento (AS) têm recebido forte atenção de pesquisadores e da indústria, uma vez que o conteúdo compartilhado pelos usuários fornece informações valiosas sobre o sentimento e interesse dos usuários.

De modo geral, a AS é uma tarefa de categorização com o objetivo de classificar o sentimento de textos escritos em linguagem natural, considerando a sua polaridade, se positiva ou negativa [42]. No contexto das redes sociais, a AS introduz uma série de desafios. Entre eles está informalidade da escrita. Neste tipo de cenário, usuários se expressam em um formato de escrita livre, chamado “internetês”, onde é comum encontrar erros ortográficos, ambiguidades e frases sintaticamente mal formadas. Com objetivo de contornar esse problema, vários esforços são dedicados na etapa de preparação de dados [50, 25, 52], sendo uma tarefa importante para melhorar a representatividade dos dados e reduzir o impacto do informalismo.

Tradicionalmente, pesquisas em AS têm focado em algoritmos especializados para processar dados em modo *offline*, geralmente utilizando pequenas quantidades de dados e distribuição estacionária. Entretanto, o Twitter é um ambiente dinâmico, que gera mensagens em fluxo contínuo, com dados que podem chegar em alta velocidade e sobre uma distribuição que pode mudar ao longo do tempo [21]. Mudanças na distribuição dos dados estão associadas a um fenômeno chamado *Concept Drift*, sendo um dos principais tópicos relacionados à classificação em Fluxo Contínuo de Dados (FCD). Mesmo que essas mudanças sejam esperadas, em várias aplicações o tipo de mudança é desconhecido pelo sistema. Em ambientes reais, alterações podem ocorrer na forma de ruídos ou oscilações, o que gera falsos positivos e reduz a capacidade preditiva do classificador. Embora algoritmos incrementais sejam capazes de incorporar novos dados e se adaptar às mudanças no ambiente, detectores de mudança fornecem informações sobre potenciais mudanças na distribuição dos dados e são capazes de reagir e gerar alertas sobre mudanças abrup-

tas [23]. Nesse sentido, várias pesquisas têm sido dedicadas ao tratamento de *Concept Drift* usando detectores de mudança [65, 54, 9, 22], sendo ainda um problema em aberto, gerando contínuos estudos em tarefas de classificação em FCD.

Além do *Concept Drift*, um tipo especial de mudança ocorre em decorrência da mudança no espaço de atributos. Chamado de *Feature Drift* [31], trata-se de um problema onde novos atributos relevantes surgem ao longo do fluxo e atributos conhecidos se tornam irrelevantes, exigindo a construção de um espaço dinâmico de atributos. Katakis *et* al [31] foram pioneiros a introduzir problema de *feature drift* em FCD. Os autores propuseram um *framework* que combina dois componentes principais: um método incremental que armazena e seleciona os atributos mais relevantes ao longo do tempo, e um classificador incremental, que utiliza os atributos selecionados para atualizar o modelo. Mais recentemente, uma abordagem foi proposta por Wenerstrom [66], que introduz um comitê de classificadores que atualiza o espaço de atributos, considerando os termos mais relevantes no fluxo.

Embora *feature drift* seja um problema que afeta diretamente a qualidade de um algoritmo de classificação, esse tema não tem sido explorado no contexto do Twitter, sendo uma tarefa apenas explorada em domínios tradicionais de Mineração de Texto, como categorização de textos e detecção de *spam*.

1.1 Objetivos

O objetivo geral deste trabalho é melhorar o processo de Análise de Sentimento em ambientes de FCD. Utilizando dados reais do Twitter, deseja-se implementar um algoritmo capaz de classificar o sentimento de textos informais, mantendo a qualidade preditiva e a aptidão para reagir de forma efetiva às mudanças no fluxo de dados.

Além do objetivo geral, os seguintes objetivos específicos foram explorados durante o desenvolvimento deste trabalho:

- Detectar e reagir de forma eficaz a mudanças na distribuição dos dados;
- Manter um espaço dinâmico de atributos, monitorando a ocorrência de novos atributos relevantes;
- Manter um comitê de classificadores incremental, capaz de incorporar novos dados e evoluir em conformidade com a distribuição mais recente.

1.2 Estrutura do Trabalho

Este trabalho está dividido em 7 capítulos, organizados da seguinte forma: o Capítulo 2 apresenta conceitos e características referentes a AS no Twitter. O Capítulo 3 elenca os fundamentos e técnicas empregadas em tarefas de classificação em FCD. O Capítulo 4 apresenta os trabalhos relacionados a esta pesquisa. O Capítulo 5 apresenta o algoritmo SENTIMENTSTREAM. O Capítulo 6 apresenta os resultados experimentais e, o Capítulo 7 destaca as conclusões e trabalhos futuros.

2. ANÁLISE DE SENTIMENTOS NO TWITTER

Este capítulo tem como objetivo elencar os principais aspectos e características relacionados a AS. Além disso, ao longo das próximas seções, são apresentados desafios e técnicas empregadas no pré-processamento e na representação de textos informais.

2.1 Visão Geral

Opiniões têm forte influência sobre o comportamento humano, sendo fundamentais para a maioria das atividades humanas. Decisões simples como comprar algum produto ou definir um local para jantar são frequentemente estimuladas pela opinião de outras pessoas, considerando o sentimento e a experiência obtida nessas atividades [19, 42]. Uma opinião é basicamente composta por dois elementos principais: um *alvo* e um *sentimento* sobre este alvo. Um alvo pode ser representado por uma entidade, indivíduo, tópico ou marca. Já um sentimento é representado por elementos subjetivos, tais como emoções, avaliações e experiências [42].

A AS, também conhecida como Mineração de Opinião, é uma tarefa de classificação com objetivo de categorizar textos, identificando seus sentimentos, avaliações, atitudes e percepções relacionadas a um alvo [42]. De modo geral, um documento d é representado através da quíntupla (e,a,h,t,s), onde **e** é o nome do alvo; **a** é o aspecto de **e**; **h** é o titular da opinião e **t** é o tempo em que o sentimento foi expresso. Um sentimento **s** pode ser classificado em classes discretas: positivo, negativo ou neutro, ou então, dentro de uma escala com diferentes níveis de intensidade, por exemplo, de -1 a 1 [42].

Um processo de AS pode ser definido de forma genérica através das seguintes etapas: (1) Coleta de documentos sobre um determinado alvo, (2) pré-processamento dos documentos, (3) classificação dos documentos e, (4) visualização dos resultados. A Figura 2.1, apresenta uma arquitetura genérica para a tarefa de AS, proposta em [19].

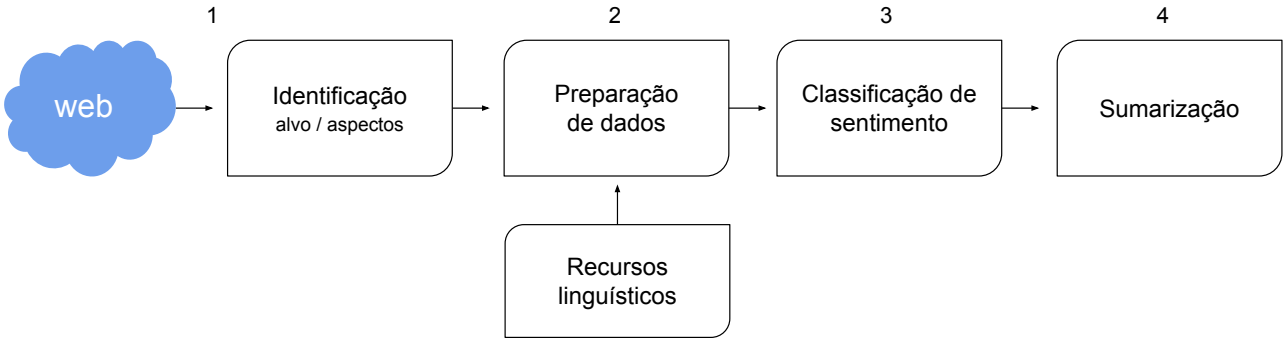

Figura 2.1 – Arquitetura genérica de um sistema para AS [19].

Ao longo do tempo, portais de *reviews* se tornaram um alvo atrativo para AS, sendo um espaço valioso para aquisição de conhecimento a partir de textos subjetivos [19]. Além disso, com o rápido crescimento de redes sociais como Twitter e Facebook, a AS evoluiu e tornou-se um importante tópico de pesquisa para diversas aplicações. Atualmente muitos usuários utilizam a web como um espaço *online*, compartilhando opiniões sobre diferentes assuntos de seu interesse. Sendo a internet um espaço livre e informal, os tópicos abordados nas redes sociais vão bem além de comentários pessoais, mas sim, englobam assuntos emergentes e de interesse da sociedade. Nesse sentido, pesquisas recentes têm direcionado esforços para melhorar o processo de AS, desenvolvendo algoritmos apropriados para lidar com dados sociais.

2.2 Desafios

A AS é uma área que nos últimos anos tornou-se um importante tópico de pesquisa em Aprendizado de Máquina (AM) e Processamento de Linguagem Natural (PLN). Muito disso se deve especialmente aos desafios e dificuldades envolvidos nesse processo. Entre esses desafios, destacamos três: a informalidade das mensagens, a alta dimensionalidade dos dados, e a capacidade de processar grandes quantidades de textos em tempo real.

Em tarefas de classificação de texto, grande parte do tempo é empregado na etapa de pré-processamento e melhoria dos dados. Além disso, com a popularização das redes sociais *online*, pesquisas em AS se deparam com um problema em particular: o "internetês". No Twitter, por exemplo, é comum que mensagens contenham erros gramaticais, abreviações, contrações e outras alterações na estrutura textual. Essas variações prejudicam a capacidade de generalização do algoritmo e aumenta o espaço dimensional, exigindo que estratégias de pré-processamento apropriadas sejam empregadas.

Além da informalidade, as redes sociais caracterizam-se como um processo de FCD, gerando novas mensagem em tempo real, com alta velocidade e distribuição que pode mudar ao longo do tempo [8]. Nesse contexto, duas questões são importantes. A primeira, chamada de *concept drift*, ocorre quando o sentimento sobre o alvo analisado muda ao longo do tempo [23]. Já a segunda, conhecida como *feature drift*, é um tipo de mudança que ocorre quando atributos relevantes surgem ao longo do fluxo e atributos conhecidos se tornam irrelevantes, exigindo que o espaço dimensional seja atualizado [31]. Apesar de serem problemas distintos, ambos afetam a capacidade preditiva do classificador, sendo necessário algoritmos adaptativos para atender esses desafios.

2.3 Níveis de Análise

A classificação de sentimentos pode ser realizada em diferentes níveis de granularidade. A decisão do nível utilizado depende do grau de informação que deseja-se extrair e do tipo de opinião expressa pelos usuários. De modo geral, a AS está dividida em três níveis principais [42]:

- **Documento**: seja *d* um documento opinativo que expressa o sentimento sobre o alvo *e*, o objetivo dessa analise é determinar o sentimento geral em *d*. No nível de documento, assume-se que *d* expressa opinião sobre um único alvo, desconsiderando sentenças e aspectos que possam estar presentes no documento [2]. Essa restrição é imposta pois em alguns casos, um documento pode expressar o sentimento sobre diferentes alvos e aspectos, o que torna inapropriada a análise no nível de documento.
- **Sentença**: é natural pensarmos que um documento pode conter várias opiniões sobre um mesmo alvo. Além disso, muitas vezes deseja-se analisar cada uma das opiniões, tarefa que se torna ineficaz quando analisamos documentos com nível menor de granularidade [2]. Assim, uma alternativa apropriada é empregar a AS no nível de sentença, permitindo que o sentimento de um documento seja analisado individualmente em cada sentença.
- **Aspecto**: embora os níveis de documento e sentença sejam eficazes em alguns casos, muitas vezes eles não fornecem detalhamento suficiente sobre o sentimento do alvo analisado. Quando um produto é analisado em nível de documento e classificado como positivo, podemos perder algum ponto negativo citado no texto, o que exige que um processo mais detalhado seja adotado. Nesses casos, é adequado descer um nível e empregar a classificação no nível de aspecto, com foco na opinião sobre diferentes aspectos de um alvo específico [2].

Tweets têm como características principais o tamanho curto (no máximo 140 caracteres) e o tipo de conteúdo, normalmente informal e opinativo, expressando o sentimento sobre um único alvo. Dessa forma, nesse trabalho será empregada a AS em nível de documento, onde o objetivo é classificar a polaridade de um *tweet*, sem considerar sentenças e aspectos.

2.4 Abordagens para Análise de Sentimento

Devido aos desafios que envolvem a extração de conhecimento em dados textuais, diversas abordagens de AS têm sido desenvolvidas ao longo do tempo. De modo

geral, essas abordagens são divididas em quatro grandes grupos [62]: (1) Abordagem baseada em Dicionário, (2) Abordagem Estatística, (3) Abordagem Semântica e (4) Abordagem baseada em Aprendizado de Máquina.

1. **Abordagem baseada em Dicionário:** a abordagem baseada em Dicionário, também chamada de *Lexical* ou *Linguística*, faz uso de um léxico de sentimento (dicionário), que são compilações de palavras ou intensificadores de sentimento, associados à sua polaridade. Métodos comuns que empregam a abordagem lexical são baseados na co-ocorrência de palavras. Neste caso, para classificar um sentimento, basta identificar uma palavra que contenha sentimento (por exemplo, um adjetivo), e buscá-la no dicionário lexical. Normalmente este tipo de método apresenta bons resultados quando empregado em sentenças ou documentos curtos, justificada pela presença de uma palavra detentora de sentimento próximo ao alvo de análise [6].

2. **Abordagem Estatística:** considerando que palavras e frases opinativas são fortes indicadores de opinião, a abordagem estatística, também denominada de *não supervisionada*, está baseada na premissa onde palavras que expressam alguma opinião frequentemente são encontradas juntas em uma sentença, ou seja, se uma palavra ocorre frequentemente ao lado de uma palavra positiva, então provavelmente ela também será positiva [6].

3. **Abordagem Semântica:** de forma semelhante a abordagem Estatística, na abordagem semântica a polaridade é calculada através de uma medida de distância entre dois termos, onde essa distância pode ser avaliada através da qualidade no relacionamento entre os mesmos (e.g. sinônimos e antônimos) [43, 6].

4. **Abordagem baseada em Aprendizado de Máquina:** o Aprendizado de Máquina é uma sub-área da Inteligência Artificial com objetivo de, com base em dados, automatizar o processo de descoberta de conhecimento através de técnicas computacionais. Na AS, normalmente são empregadas técnicas de classificação, onde tradicionalmente dois passos são realizados [62]: (1) gerar um modelo preditivo a partir dados rotulados e (2) classificar a polaridade de dados não rotulados utilizando o modelo gerado. Pesquisas tradicionais de AS normalmente utilizam técnicas de classificação como *Naïve Bayes*, SVM e *Maximum Entropy*.

Considerando os bons resultados em tarefas de Análise de Sentimento, neste trabalho, propomos um abordagem baseada em Aprendizado de Máquina, onde nosso objetivo construir um comitê de classificadores para AS em FCD. Através desse comitê, buscamos obter melhores resultados preditivos do que abordagens tradicionais, que utilizam apenas um modelo para realizar suas predições.

2.5 Pré-processamento de Textos

A tarefa de pré-processamento inclui uma série de rotinas, processos e métodos necessários para melhorar a qualidade dos dados e alcançar melhores resultados preditivos. Na mineração de textos, essa tarefa é ainda mais importante, pois a maioria das técnicas de AM não processam textos em sua forma original, sendo necessário convertê-los para uma representação que seja manejável por algoritmos de classificação [20].

O pré-processamento pode ser explorado de diferentes maneiras, dependendo do tipo de informação que os dados fornecem e sua estrutura original. Uma diferença importante entre os textos de mídias sociais e mídias tradicionais (jornais e portais de notícias) é a variação na qualidade do conteúdo. No *Twitter*, por exemplo, as mensagens estão limitadas à 140 caracteres. Devido a esta limitação, textos são curtos e muitas vezes contém abreviações, gírias, erros gramaticais e outras estruturas textuais que são características da informalidade das redes sociais. Dessa forma, textos informais fazem do pré-processamento uma tarefa complexa e desafiadora, consumindo mais tempo do que o tratamento de textos de mídias tradicionais [3].

A Tabela 2.1 apresenta um exemplo comparativo entre uma sentença escrita em português formal e outras três variações expressas usando linguagem informal. Na tabela é possível identificar diversas variações na estrutura textual, tais como abreviações “c/”, palavras alongadas “BRIGADEIROOOOOO”, *emoticons* “:PPP” e erros gramaticais “cum”, que são estruturas facilmente encontradas nas redes sociais. Além de exigir maior esforço na tarefa de pré-processamento, sendo necessário “traduzir” as mensagens para o português formal, tais estruturas são úteis e fornecem informações valiosas sobre a subjetividade e o sentimento das mensagens. Dessa forma, através de textos informais é possível tanto mensurar o grau de subjetividade de uma mensagem, como capturar seu sentimento.

Tabela 2.1 – Exemplo de uma sentença escrita em linguagem formal e três variações possíveis utilizando a linguagem informal.

Formal	Estou com desejo de comer brigadeiro.
Informal	Estou c/ desejo d brigadeiro :PPP
Informal	to c 10sejo d brigadeiro :PPP
Informal	TO CUM DESEJO DI BRIGADEROOOOOO

Tendo em vista os desafios associados ao tratamento de textos informais, diversas abordagens de AS exploram a informalidade e destinam seus esforços no pré-processamento [50, 55, 52, 25]. A Figura 2.2 apresenta um esquema genérico da tarefa de pré-processamento e introduz algumas possibilidades existentes nessa tarefa. Seja

$\mathcal{D}$ um conjunto de documentos, inicialmente é empregado um conjunto de técnicas visando explorar o informalismo e extrair termos distintos a partir dos documentos. Essa etapa é utilizada para obter um vocabulário de termos relevante para construção de uma representação estruturada de dados.

Além do tratamento dos termos, um etapa importante é a definição de uma abordagem de representação. Em tarefas de AS é comum que sejam empregados unigramas [64, 8, 10, 24]. Entretanto, algumas pesquisas exploram bigramas e estratégias híbridas, por exemplo, usando unigramas + bigramas + *Part of Speech*, o que auxilia para extrair mais informação sobre cada documento.

Em redes sociais, além de utilizar palavras como atributos, é comum que outros elementos como *pontuações*, *hashtags*, *emoticons* sejam inseridos no espaço dimensional. Esses elementos são importantes, pois fazem parte do vocabulário da linguagem expressa nas redes sociais. Por fim, os textos são convertidos em um vetor de espaço n-dimensional, considerando o modelo de representação adotado e o vocabulário obtido através do conjunto de dados. Normalmente esse tipo de representação é numérica, podendo ser binária, por frequência ou ponderada.

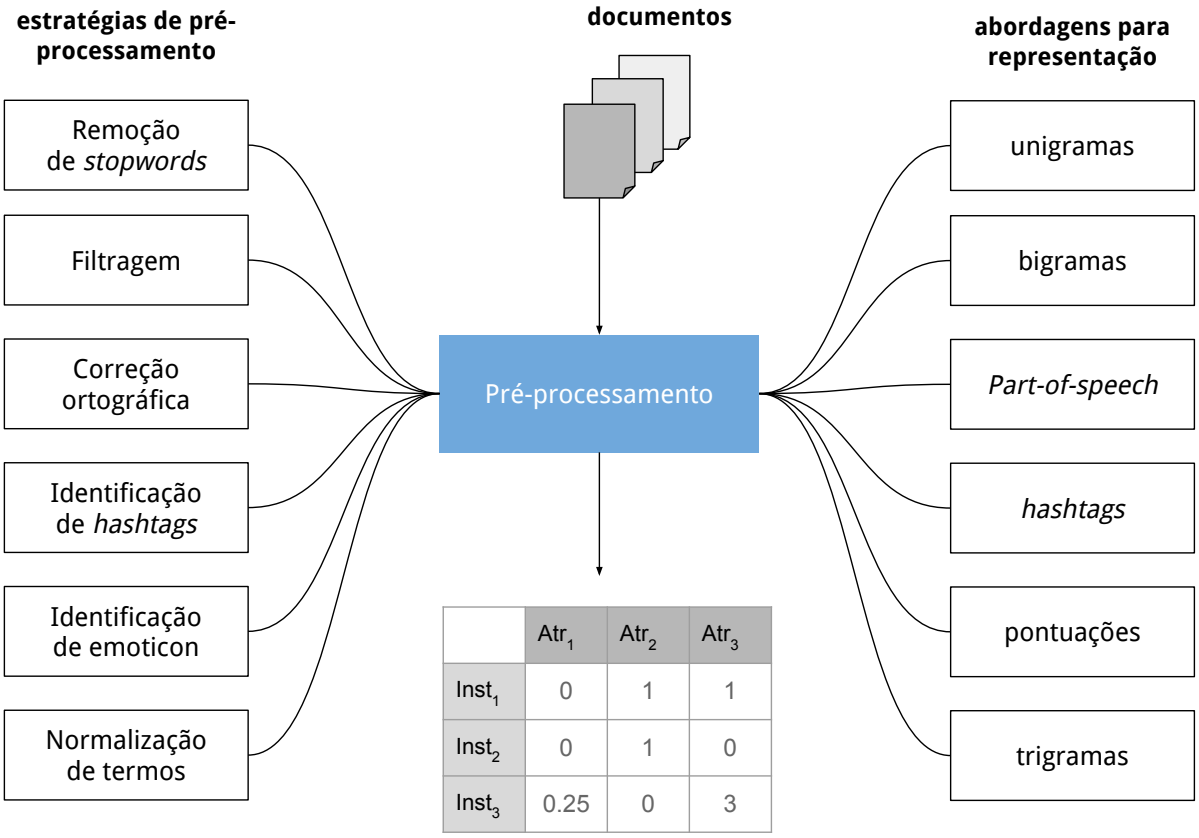

Figura 2.2 – Esquema do pré-processamento de dados informais.

2.5.1 Representação de dados não estruturados

Durante a tarefa de pré-processamento, tipicamente é necessário converter as mensagens em um modelo de representação estruturado, capaz de ser manipulável e processado por algoritmos de classificação. Na AS, a forma mais comum de gerar um modelo estruturado é transformá-lo em um vetor numérico esparso, ideal para ser processado através de operações algébricas. Essa representação, chamada de *bag-of-words* (BOW), considera os dados como espaço n-dimensional, onde n é o número de atributos e cada dimensão é representado por um atributo e seu respectivo valor. Além disso, BOW representa cada mensagem como um vetor de atributos independentes, desconsiderando a ordem em que os termos ocorrem originalmente no documento [2].

A forma mais simples de construir um modelo BOW é através de uma representação binária. Nessa abordagem, cada atributo recebe valor 1, caso esteja presente no documento e 0, caso contrário. Além do modelo binário, duas abordagens populares para representação são TF e TF-IDF. TF (*term frequência*), corresponde a frequência em que um termo ocorre em cada documento. Já TF-IDF (*term frequency-inverse document frequency*), corresponde a uma versão ponderada de TF. Sua principal vantagem é ponderar a importância do termo por um fator que indica sua representatividade no conjunto de documentos, e não apenas no próprio documento em que ele ocorre [2]. TF-IDF pode ser calculado através da Equação 2.1 [38].

$$tfidf(t, d) = tf(t, d) \cdot \log_2 \frac{n}{df(t)} \tag{2.1}$$

onde $tf(t, d)$ é a frequência do termo em cada documento (o número de vezes que t ocorre em um documento). $df(t)$ corresponde a frequência do documento (o número de documentos contendo o termo t) e n é o número de documentos do conjunto de dados.

Quando aplicada no Twitter [8, 50, 55, 52], a AS normalmente emprega a abordagem binária [8, 50], uma vez que textos do Twitter são curtos e normalmente contém termos que ocorrem apenas uma vez no texto. TF-IDF é mais apropriada em situações onde são considerados documentos de tamanhos distintos. Aplicando TF-IDF, é possível evitar que termos muito frequentes predominem sobre a decisão preditiva e prejudiquem a capacidade do algoritmo generalizar corretamente os dados.

Em tarefas de AM, a definição de um modelo de representação apropriado é um fator importante para alcançar qualidade preditiva. Além do modelo BOW tradicional, criado a partir de técnicas como tokenização e remoção de *stopwords*, outras estratégias são utilizadas para melhorar a representatividade do texto. Entre elas estão: n-gramas, POS (*part-of-speech*) e intensificadores de sentimento, elementos típico de textos subjetivos. A seguir são descritas cada uma dessas estratégias.

- **N-gramas:** n-gramas são unidades formadas por uma sequência contígua de *n* termos. Ao contrário do modelo BOW tradicional, n-gramas permitem capturar maior contexto dos textos, evitando a total independência entre os termos gerados pelos unigramas. Abordagens que empregam n-gramas normalmente utilizam bigramas ou trigramas para representar os atributos, ou seja, formam sequências de dois ou três termos no máximo. Entretanto, trigramas não apresentam resultados eficazes, sendo bigramas a forma mais comum de n-gramas empregada em AS [52].

- ***Part of Speech (POS):*** também chamado de *Tagger*, caracteriza-se pelo processo de anotação de palavras, ou seja, busca determinar uma classe gramatical para um determinado termo, considerando o contexto em que ele aparece no texto. De modo geral, POS é um algoritmo que divide as palavras em categorias gramaticais, tais como adjetivos, advérbios, verbos ou pronomes, fornecendo informação sobre a semântica dos termos [20]. Em AS, é comum o uso de POS [7, 34], uma vez que esse processo permite identificar palavras consideradas como indicadores de subjetividade. Entre esses termos, os mais representativos normalmente são adjetivos e advérbios [51]. Além disso, vários estudos em AS exploram abordagens híbridas, empregando técnicas de AM e POS, buscando melhores resultados no processo de classificação [7, 34].

- **Intensificadores de sentimento:** além de considerar palavras como atributos, em redes sociais é comum que outros elementos como *pontuações*, *hashtags*, *emoticons* sejam inseridos no espaço dimensional [25]. Além disso, muitos usuários expressam seus sentimentos usando palavras alongadas ou em caixa alta, exigindo que o pré-processamento capture e normalize esse tipo de informação.

2.5.2 Seleção de atributos

Antes de realizar o processo de classificação, uma das principais etapas para aquisição de um modelo preditivo adequado é a seleção de atributos. De modo geral, a seleção de atributos caracteriza-se pelo processo de identificar um subconjunto de atributos relevantes para a tarefa de classificação. Além de ser importante em diversas tarefas de mineração de dados, esse processo torna-se necessário em tarefas de classificação textual, uma vez que esse tipo de cenário pode gerar um grande volume de atributos, muitos deles irrelevantes para um modelo de Aprendizado de Máquina [2].

As técnicas mais comuns empregadas para redução de atributos em problemas de mineração de texto são: *stemming* de palavras e remoção de *stopwords*. Na remoção de *stopwords*, termos considerados irrelevantes são descartados uma vez que não contribuem para a classificação. Esses termos normalmente são artigos, conjunções e

pronomes, pois aparecem com muita frequência no conjunto de treinamento e ao mesmo tempo não têm boa correlação com alguma classe alvo. Além disso, em alguns casos, termos raros também são considerados irrelevantes, uma vez que podem ser considerados como ruídos. Já no processo de *stemming*, a ideia principal é normalizar e converter para uma único atributo termos que possuem o mesmo significado, mas que são escritos de forma flexionada.

Além das técnicas de *stemming* e remoção de *stopwords*, é comum filtrar atributos com base em sua relevância, relacionando a alta correlação entre cada termo e as classes alvo consideradas no problema. Na literatura de mineração de textos, diversas técnicas têm sido empregadas para auxiliar nesse processo [68], entre elas: Entropia, Estatística χ^2 e Informação Mútua são as mais utilizadas.

2.5.3 Entropia

Entropia é um critério externo de validação, ou seja, um método que utiliza o conhecimento prévio sobre os dados para determinar a desordem ou impureza dos atributos a respeito das classes do problema. Dado um conjunto de treino, a Entropia utiliza a probabilidades *a posteriori* da classe c_i para determinar o quanto o atributo t contribui para decidir o sentimento da classe. Segundo Shannon e Elwood [56], a Entropia de um termo pode ser obtida através da Equação 2.2.

$$Entropia(t) = -\sum_{i=1}^{T} p(c_i|t) \cdot \log_2 p(c_i|t) \tag{2.2}$$

onde T é o número de classes consideradas no problema e $p(c_i|t)$ é a probabilidade do termo t pertencer a classe c_i. Os valores são normalizados em uma escala de 0 a 1. Valores próximos de 1 indicam que o termo possui ocorrências em diferentes classes. Quanto maior o valor, mais uniforme será a presença de t em todas as classes, indicando que esse termo não contribui para classificação. Por outro lado, valores baixos de Entropia indicam que t ocorre com mais frequência em apenas uma classe, sendo assim, significativa para classificação.

2.6 Considerações Finais

Este capítulo introduziu os fundamentos básicos e os desafios relacionados à AS no Twitter, aplicação alvo desta pesquisa. Ao longo do capítulo foram apresentadas técnicas empregadas no pré-processamento, focando em duas questões principais: o trata-

mento de textos informais e as estratégias utilizadas para transformar dados textuais em um modelo de representação estruturado.

O próximo capítulo tem como objetivo apresentar os fundamentos do aprendizado em FCD, focando em técnicas empregadas na construção de algoritmos adaptativos.

3. APRENDIZADO EM FLUXO CONTÍNUO DE DADOS

Nas últimas décadas, pesquisas e aplicações em AM têm focado no desenvolvimento de algoritmos especializados para processar dados em *batch* (*offline*), geralmente utilizando pequenas quantidades de instâncias, armazenados em bancos de dados convencionais e sobre uma distribuição que não muda ao longo do tempo [21]. Nesse tipo de cenário, considera-se que todos os dados de treino estão disponíveis para o algoritmo, que produz um modelo de decisão após processar múltiplas vezes os dados, gerando assim um conjunto de exemplos de treino que atuam sobre uma distribuição de probabilidade estacionária [21].

Por outro lado, problemas do mundo real são dinâmicos, contendo dados que chegam continuamente, em alta velocidade e com distribuição que pode mudar ao longo do tempo. Esse tipo de cenário, denominado como Fluxo Contínuo de Dados [21], caracteriza-se por um processo estocástico, onde eventos independentes ocorrem continuamente ao longo do tempo. Em nosso cotidiano, operações simples como utilizar o cartão de crédito gera de forma automática uma infinidade de dados, que são factíveis de serem analisados em tempo real através de algoritmos *online*. Da mesma forma, redes sociais como *Twitter e Facebook*, são exemplos de valiosas fontes de dados, uma vez que produzem grandes volumes de textos, imagens e vídeos gerados em um ambiente que pode evoluir ao longo do tempo.

3.1 Características

Formalmente, um FCD $\mathcal{S}$ é uma sequência massiva de instâncias $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_N$, isto é, $\mathcal{S} = \{\mathbf{x}_i\}_{i=1}^{N}$ potencialmente infinita ($N \longrightarrow \infty$) e gerada ao longo do tempo. Cada instância é representada por um vetor n-dimensional de atributos, pertencente a um espaço Ω que pode ser categórico, contínuo ou misto. As instâncias são associadas a um marcador de tempo (*timestamp*) que pode ser explícito, quando fornecido pela fonte que gerou a instância, ou implícito, quando obtido pelo sistema que a coletou [58]. Babcock *et* al. [4] elenca algumas das principais diferenças entre aprendizado *offline* e o aprendizado em FCD:

- os dados chegam continuamente, muitas vezes em alta velocidade;
- o sistema não tem controle da ordem em que os elementos chegam através do fluxo;
- o fluxo é considerado infinito, portanto, uma pequena quantidade de dados deve ser armazenada em memória.

Devido ao grande volume de dados gerado e sua natureza dinâmica, o aprendizado em FCD requer que novas estruturas de dados e técnicas de mineração sejam desenvolvidas [21]. Um dos grandes desafios desse cenário é a capacidade de manter a qualidade preditiva ao longo do tempo, sendo robusto o suficiente para lidar com mudanças na distribuição dos dados. Nesse sentido, uma propriedade importante é a construção de algoritmos capazes de incorporar novos dados constantemente, já que mudanças no ambiente de processamento podem ocorrer ao longo do tempo e o processo incremental permite que o algoritmo generalize os dados de acordo com a distribuição mais recente.

Além da incrementalidade, algoritmos devem ser capazes de se adaptar na ocorrência de *concept drift*. Enquanto abordagens incrementais atualizam o modelo preditivo de forma lenta, onde um novo conceito é aprendido incrementalmente ao longo do tempo, algoritmos adaptativos fazem uso de métodos reativos para detecção de mudança, o que torna o algoritmo mais robusto para atuar em ambientes dinâmicos e mais eficaz no aprendizado de novos conceitos [23].

Nesse capítulo é apresentada uma revisão de conceitos e métodos empregados na construção de algoritmos para o aprendizado em FCD. Baseado na taxonomia proposta por Gama *et* al. [23], seis aspectos relacionados a esta pesquisa são observados: (i) mecanismos de esquecimento, (ii) detecção de mudança, (iii) espaço dinâmico de atributos, (iv) modo de aprendizado, (v) comitês de classificadores e (vi) avaliação de classificadores em FCD.

Ao longo das próximas seções, cada um dos componentes é explorado, caracterizando sua importância e como cada um atua na construção de um algoritmo adaptativo.

3.2 Mecanismos de esquecimento

O aprendizado em FCD não exige apenas a atualização do modelo preditivo com instâncias mais relevantes, mas sim, necessita que dados antigos e/ou irrelevantes sejam descartados ao longo do tempo. Devido às restrições computacionais, interessa manter em memória apenas os dados relevantes. Seguindo essa ideia, a maioria dos algoritmos adaptativos consideram que dados mais recentes são mais importantes para o aprendizado, uma vez que eles representam a distribuição mais recente dos dados [23].

A forma mais comum e disseminada para manter uma quantidade factível de dados em memória são as janelas de eventos [21]. Janelas, também chamadas de mecanismos de esquecimento, definem a quantidade e como os dados são armazenados ao longo do fluxo. Babcock *et* al. [4] apresentam dois modelos básicos de janela:

Janela deslizante. São janelas definidas pelo número de dados que estão armazenando. Também chamada de *Sliding window*, esse tipo de janela pode ser de tamanho fixo ou variável. Quando o tamanho é fixo, ela segue uma estrutura baseada em uma fila (FIFO

- *first in first out*). Conforme ilustra a Figura 3.1 (a), quando a janela estiver cheia, a cada nova instância $\mathbf{x}_i$ que chega no fluxo, a instância mais antiga é descartada. Janelas de tamanho variável não definem um tamanho fixo de dados. Normalmente o tamanho da janela é definido através de indicadores de detecção de mudança, que determinam o quanto a janela pode e deve crescer [21].

Janela de marcação. Também chamada de *Landmark window*, essa abordagem gera uma janela de tamanho variável, definida por um marcador de tempo. Conforme ilustra a Figura 3.1 (b), esse marcador armazena as instâncias que que chegam dentro de um período especificado, descartando os dados logo após eles serem processados. Diferente da *sliding window*, janelas de marcação não deslizam ao longo do fluxo, mas sim, iniciam uma nova janela sempre que a janela anterior estiver preenchida.

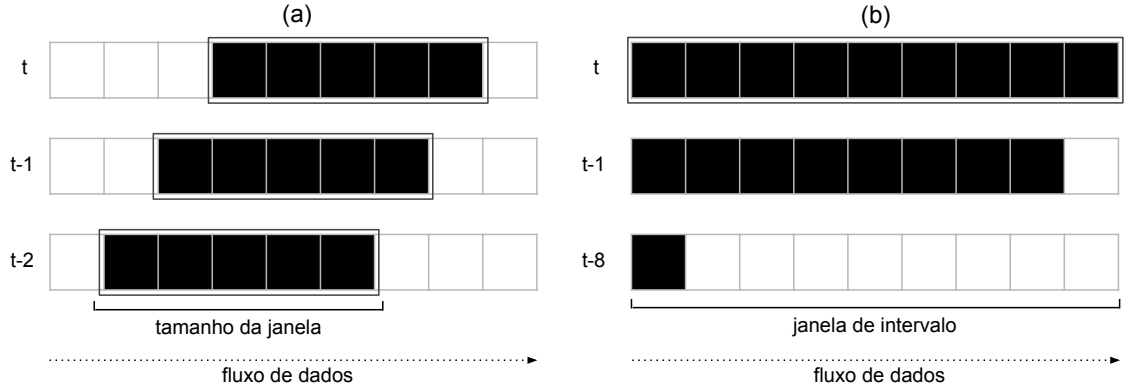

Figura 3.1 – Modelos de janela (a) deslizante e (b) de marcação.

3.3 Detecção de mudança

No *Twitter*, assim como a maioria dos ambientes do mundo real, algoritmos devem ser capazes de identificar mudanças existentes no ambiente, controlar o seu comportamento e adaptar o modelo preditivo em conformidade com a distribuição mais recente dos dados [23].

Em problemas de aprendizado supervisionado, mudanças na distribuição que gera novos dados dentro do fluxo são chamadas de *concept drift*. Formalmente, um *concept drift* entre dois momentos t_0 e t_1 pode ser definido através da Equação 3.1 [23].

$$\exists X : p_{t_0}(X, y) \neq p_{t_1}(X, y) \tag{3.1}$$

onde p_{t_0} denota a probabilidade conjunta no tempo t_0 entre o conjunto de variáveis X e a classe alvo y. A mudança nos dados pode ser caracterizada pela alteração que ocorre entre os componentes dessa relação. Kelly *et* al. [32] apresentam três formas em que um *concept drift* pode ocorrer:

- a probabilidade a *priori* da classe $P(y)$ pode mudar;
- a probabilidade condicional da classe $P(X|y)$ pode mudar;
- a probabilidade *a posteriori* das classes $P(y|X)$ pode mudar, afetando a predição.

Nesse contexto, duas implicações são importantes: (i) se a distribuição dos dados $P(y|X)$ muda e afeta a decisão preditiva e (ii) se as mudanças são visíveis a partir da distribuição de dados sem conhecer os verdadeiros rótulos (ou seja, mudanças em $P(X)$. Sob uma perspectiva preditiva, apenas as alterações que afetam a predição necessitam de adaptação. Dessa forma, é possível dividir uma mudança em duas classes principais [23]:

1. Real *concept drift*: refere-se a mudanças em $P(y|X)$. Tais alterações podem ocorrer com ou sem alteração em $P(X)$;
2. *virtual drift*: ocorre se a distribuição dos dados de entrada muda (isto é, $P(X)$), sem afetar $P(y|X)$.

Detecção de *concept drift* em ambientes dinâmicos é uma importante tarefa do aprendizado *online*, uma vez que as mudanças podem se manifestar de diferentes formas, como mostra a Figura 3.2. Normalmente mudanças têm comportamento *abrupto*, quando a distribuição muda de forma rápida. *incremental ou gradual*, onde as mudanças ocorrem lentamente e ao longo do tempo, ou *recorrentes*, representando mudanças que ocorrem repetidamente ao longo do fluxo de processamento [23]. Além destes comportamentos, *outliers* e *ruídos* não são considerados verdadeiras mudanças. *Outliers* representam eventos raros que ocorrem ao longo do fluxo e indicam anomalias ou comportamentos estranhos que ocorrem de forma aleatória. Apesar de não exigir adaptabilidade do modelo de decisão, sua detecção é importante em vários domínios [36]. Já os *ruídos*, são flutuações insignificantes, mas que em alguns casos pode prejudicar e influenciar na detecção de mudanças, causando falsos positivos ao longo do processamento.

Devido ao dinamismo das aplicações do mundo real, modelos preditivos devem estar preparados para se adaptar as contínuas mudanças dos dados, uma vez que essas mudanças são esperadas mas não podem ser identificadas previamente. Além disso, algoritmos devem ser capazes de detectar rapidamente novos conceitos e atuar com pequenas sequencias de instâncias, respeitando os requisitos do aprendizado *online*.

3.3.1 Aplicações

Devido as características do mundo real e as propriedades específicas de cada tipo de cenário, diversas aplicações discutem e endereçam problemas de *concept drift*.

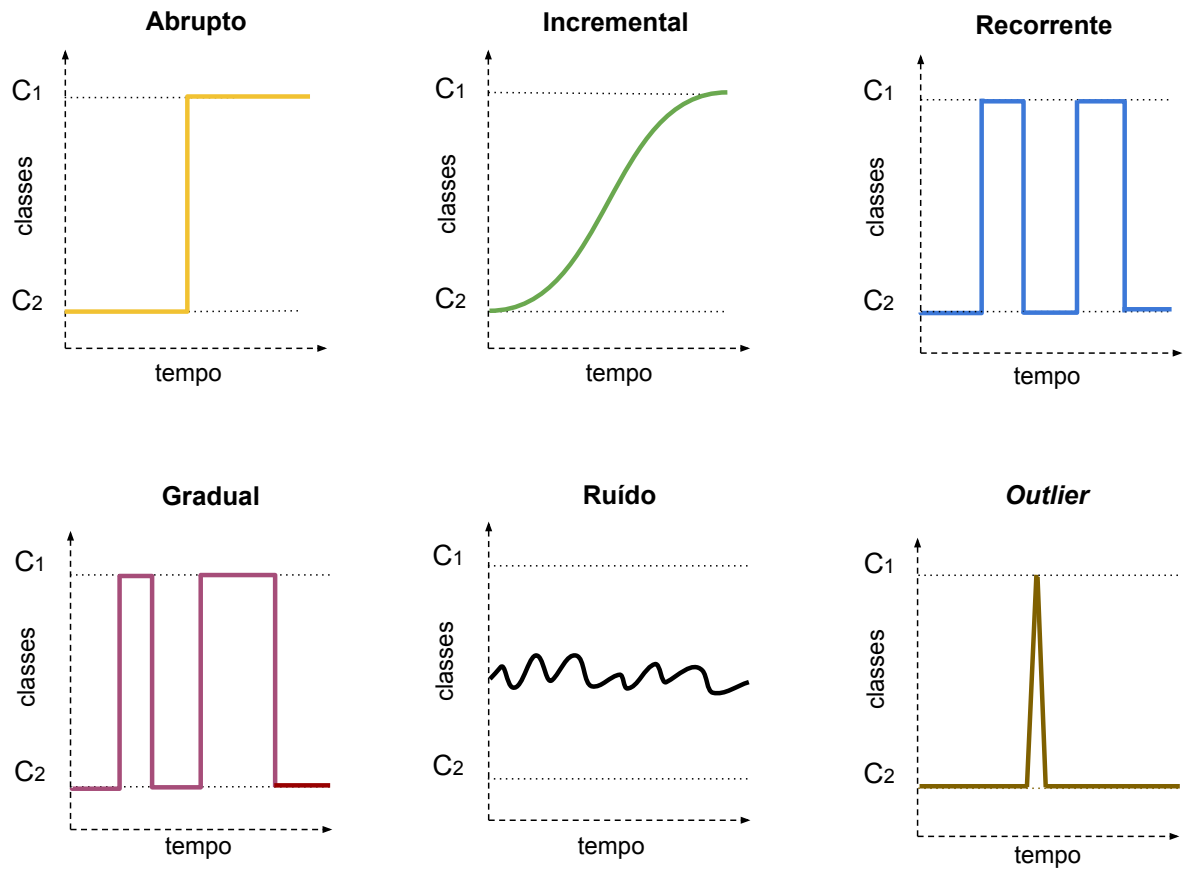

Figura 3.2 – Tipos de mudança em FCD.

Em várias cenários como detecção de anomalias [53], filtragem de *spam* [39] ou categorização de textos [15], o processo preditivo requer atenção especial na identificação de mudanças no ambiente de processamento. Aplicações que necessitam adaptação podem ser organizadas em três categorias [14]: (i) Assistentes pessoais, (ii) Monitoramento de sistemas e (iii) Suporte de decisões.

- **(i) Assistente Pessoais -** Técnicas de aprendizado em FCD têm sido diretamente relacionadas a personalização e organização de dados em diversos domínios. Nos últimos anos, aplicações para classificação de textos tornaram-se um popular tópico AM, tratando de problemas como filtragem de e-mails [15], Análise de Sentimento [8], detecção de *spam* [39] e categorização de notícias [30].

  Em tarefas de categorização de texto, a principal fonte de mudanças na distribuição dos dados é a filtragem de *spam*. Esse tipo de problema envolve diversos desafios: um deles relacionado a mudança do conteúdo de e-mails. Nesse cenário, sistemas que encaminham *spams* alteram frequentemente as palavras e a linguagem utilizada, visando burlar a filtragem e fazer com que o usuário reconheça como um e-mail real. Além disso, conforme é o foco deste trabalho, redes sociais como Twitter

tornaram-se valiosas fontes de conteúdo opinativo. Nesse tipo de plataforma, mudanças na distribuição dos dados é muito comum, uma vez que a opinião do público sobre determinado tópico pode mudar constantemente, de acordo com a evolução do assunto pautado.

- **(ii) Monitoramento de sistemas.** são grandes quantidades de fluxos de dados que precisam ser processados e analisados em tempo real. Normalmente esse tipo de problema trata de dados desbalanceados, contendo grandes quantidades de ruídos, o que torna o problema mais desafiador. Uma típica tarefa nesse domínio é distinguir comportamentos esperados daqueles não esperados pelo sistema. Aplicações envolvem segurança em redes de computadores [53], detecção de fraudes [27] e monitoramento de veículos [47].

- **(iii) Suporte de decisões -** está relacionado ao diagnóstico e avaliação da legitimidade de um processo que esteja sendo realizado. Nesse tipo de cenário normalmente a decisão ocorre de forma mais demorada, normalmente com um número limitado de dados e sem a necessidade de realizar um diagnóstico em tempo real. O principal compromisso nesse tipo de tarefa é a boa acurácia. Aplicações relacionadas à suporte de decisão incluem análise de finanças [35] e aplicações biomédicas [61].

3.4 Espaço Dinâmico de atributos

Quando aplicada em FCD, a AS introduz uma série de dificuldades, entre elas a alta dimensionalidade, ausência de estrutura e muitas vezes *concept drift*. Além dessas características, essa seção explora um tipo especial de mudança, chamada *feature drift* [30]. Trata-se de um problema onde novos atributos relevantes podem surgir ao longo do fluxo e atributos conhecidos podem se tornar irrelevantes. Embora *concept drift* e *feature drift* sejam problemas distintos, ambos afetam o poder preditivo do classificador.

Feature drift é um problema característico da Mineração de Textos [31]. No Twitter, por exemplo, milhões de mensagens são compartilhados diariamente, gerando um vocabulário extremamente esparso e indisponível no início do fluxo. Além disso, tópicos dentro de um domínio de discurso podem mudar ao longo do tempo, fazendo com que novas palavras se tornem relevantes e passem a descrever mais efetivamente o contexto mais recente.

Em problemas tradicionais de classificação, normalmente o espaço dimensional é estático [50, 8, 52]. Nesses casos, um conjunto de atributos é selecionado a partir das instâncias de treino, sendo utilizados ao longo de todo período de classificação. Em vários casos, a atualização do espaço dimensional está associada a atualização do modelo preditivo, ou seja, existe uma dependência da substituição do modelo para que exista al-

guma alteração no espaço dimensional e na representatividade do texto processado [30]. Conforme ilustra a Figura 3.3, a atualização pode ocorrer de duas formas. A primeira (lado esquerdo), considera apenas os atributos obtidos no conjunto de treino inicial, sem que haja atualização do espaço dimensional do modelo ao longo do tempo. Já a segunda (lado direito), ocorre durante os períodos de retreinamento do classificador, onde ao longo do tempo os atributos são atualizados. Embora o retreinamento modifique o espaço dimensional, essa estratégia não garante que atributos relevantes sejam adicionados, mas apenas que os atributos presentes na última janela sejam considerados.

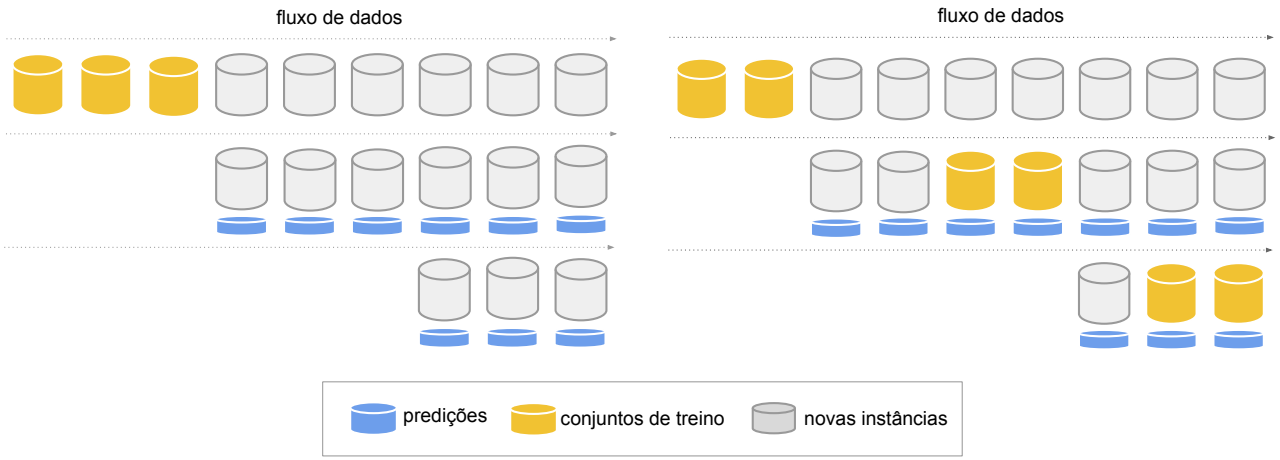

Figura 3.3 – Estratégias de aprendizado em FCD.

Além da ocorrência de novos termos, atributos que fazem parte do espaço dimensional podem se tornar irrelevantes ao longo do tempo, indicando que incrementar o espaço dimensional não é suficiente. Seja k um conjunto de atributos no tempo t_k, um *feature drift* ocorre quando no tempo t_{i+1}, existe um novo subconjunto de atributos k' relevante, onde $k \neq k'$ o suficiente para representar uma mudança no espaço dimensional.

Mudanças na relevância dos atributos afetam o poder preditivo de um algoritmo de classificação. Dessa forma, *feature drift* pode ser visto como um tipo especial de mudança no ambiente, onde existe necessidade de identificar e adaptar o espaço dimensional com novos atributos relevantes durante o processamento dos dados. Estratégias desenvolvidas para lidar com *feature drift* envolvem comitês de classificadores [66] e abordagens incrementais de aprendizado [31].

3.5 Modo de Aprendizado

Ambientes em FCD não demandam apenas processamento rápido dos dados, uma característica importante deve ser a capacidade de identificar mudanças na distribuição dos dados e adaptar o modelo preditivo para evitar o aumento do erro preditivo. Na

literatura, existem duas formas principais de gerenciar a atualização do modelo preditivo: métodos cegos e métodos informados [23].

Os métodos cegos atualizam o modelo predito em tempos regulares, sem que haja qualquer detecção de mudança. Normalmente esse tipo de abordagem utiliza um processo incremental, que atualiza o modelo sempre que existe o *feedback* para a classe verdadeira de uma instância, ou então, mantém um *batch* de instâncias na memória, e ao longo do tempo atualiza o modelo. Já o método informado é reativo, normalmente utilizando detectores de mudança para determinar a atualização do modelo preditivo [23]. Nesse método, sempre que é detectado um *concept drift*, o modelo é incrementado ou substituído, utilizando normalmente os dados mais recentes no fluxo. Ambas as abordagens são capazes de manter o modelo atualizado ao longo do tempo. Entretanto, apenas o método informado gera estimativas sobre as mudanças que ocorrem no fluxo de dados, o que é importante para observar o comportamento do ambiente. Além disso, métodos informados têm a vantagem de exigir menos processamento do que os métodos cegos, uma vez que a atualização do modelo é necessária apenas na ocorrência de *concept drift*.

3.6 Comitês de classificadores

De modo geral, comitê ou *ensemble* trata-se de um conjunto de classificadores cuja decisão individual de cada componente é combinada de alguma forma (normalmente através de um método de votação ponderada) para classificar uma nova instância [37]. O uso dos comitês é motivada por dois fatores principais: o aumento da qualidade preditiva, uma vez que podem fornecer melhores resultados em comparação ao uso de um único classificador, e a capacidade de atuar em ambientes dinâmicos, sendo capazes de aprender e evoluir em concordância com a mudança na distribuição dos dados. Além disso, para alcançar bons resultados preditivos, duas condições precisam ser atendidas: diversidade e precisão [18]. Um classificador preciso é aquele que alcança menor erro preditivo do que um classificador aleatório, ou seja, com taxa de erro menor do que 0.5 em problemas binários. Já a diversidade está relacionada a capacidade dos classificadores alcançarem erros preditivos distintos ao longo do fluxo de dados.

Para exemplificar a importância da precisão e da diversidade, considere três classificadores $\{h_1, h_2, h_3\}$ e uma nova instância de teste $\mathbf{x}$. Caso os três classificadores sejam iguais, se $h_1(\mathbf{x})$ erra a predição, $h_2(\mathbf{x})$ e $h_3(\mathbf{x})$ também erram. Porém, se o erro preditivo de cada classificador for distinto, se $h_1(\mathbf{x})$ errar, $h_2(\mathbf{x})$ e $h_3(\mathbf{x})$ podem prever corretamente a classe da instância. Dessa forma, através de uma esquema de voto majoritário, $h_2(\mathbf{x})$ e $h_3(\mathbf{x})$ têm a aptidão de determinar a predição correta de $\mathbf{x}$ [18].

Apesar de trazerem benefícios para o processo de classificação, comitês possuem alto custo computacional, uma vez que necessitam mais processamento do que um

único classificador. Quando consideramos um fluxo contínuo, com dados que chegam em alta velocidade, o uso de um classificador pode ser mais eficiente, uma vez que o comitê pode não ser capaz de processar o fluxo sem que haja perda de alguns dados ou atraso no processamento. Entretanto, caso o tempo não seja um fator crítico, o uso de comitês torna-se uma solução factível para resolver o problema [14].

3.6.1 Estratégias para o desenvolvimento de comitês de classificadores em FCD

Motivado pelos desafios envolvidos no aprendizado em FCD, Kuncheva [37] elencou 5 estratégias para o desenvolvimento de comitês que atuem ambientes dinâmicos. Quatro delas estão relacionadas à este trabalho: (a) comitês "Horse Racing", (b) atualização do conjunto de treino, (c) mudanças estruturais do comitê e (d) adaptação de atributos.

(a) Comitês "Horse Racing" Considere uma série de corridas de cavalos, onde uma pessoa (comitê de classificadores) deseja predizer o resultado de cada corrida. A pessoa possui m especialistas (membros do comitê), cada um realizando uma predição individual sobre o resultado das corridas. A cada nova corrida, a pessoa considera as predições anteriores, o erro preditivo de cada especialista e utiliza essa informação para atualizar o nível de confiança desses especialistas. Assim, a decisão preditiva de cada corrida é obtida considerando o histórico de confiança alcançado por cada especialista ao longo do tempo [14].

Proposto por Littlestone e Warmuth [41], *Weighted Majority Algorithm* (WMA) é o principal e mais representativo trabalho que emprega a estratégia "Horse racing". WMA funciona da seguinte forma [14]:

1. Dado um comitê de classificadores $\mathcal{L} = \{C_1, C_2, ..., C_m\}$. Inicializar todos os m pesos com $w_m = 1$;
2. Para uma nova instância $\mathbf{x}$, prever a classe da instância através da soma dos votos ponderados de cada classificador;
3. Sabendo a classe verdadeira de $\mathbf{x}$, utilizar o valor pré-definido $\beta \in [0, 1]$ para atualizar o peso dos classificadores do comitê. Caso o classificador erre a predição de $\mathbf{x}$, atualizar seu peso com $w_m \leftarrow \beta \cdot w_m$;
4. Continua a partir do passo 2.

De acordo com Littlestone [40], dois outros algoritmos seguem a mesma ideia de WMA: *Hegde* e *Winnow*. *Hedge* não utiliza o método de votação para definir qual a

classe será predita. Em vez disso, o classificador com peso mais alto é selecionado e a predição é realizada por meio desse classificador. Já *Winnow* segue a mesma ideia de WMA. Porém, utiliza uma estratégia diferente para atualizar os pesos. *Winnow* atualiza os pesos apenas quando o comitê erra a predição global. Caso $\mathcal{C}_m$ preveja corretamente **x**, o peso é atualizado $w_m \leftarrow \alpha \cdot w_m$, onde $\alpha > 1$ é um parâmetro de entrada do algoritmo. Quando $\mathcal{C}_m$ erra a predição, seu peso é diminuído, fazendo $w_m \leftarrow w_m/\alpha$, o que considera que $\mathcal{C}_m$ tem culpa sobre o erro do comitê [40].

Apesar de "*horse racing*" fornecer bons resultados em alguns cenários, esse tipo de estratégia não é apropriada para mineração em FCD. Isso ocorre pois os classificadores do comitê não são atualizados em nenhum estágio do processo. Dessa forma, o comitê pode se tornar ineficiente para classificação, uma vez que a ocorrência de um *concept drift* fará com que os membros do comitê generalizem de forma incorreta os dados.

(b) Atualização do conjunto de treino. Essa estratégia trata da incrementalidade dos componentes do comitê de classificação. O objetivo é manter um conjunto de classificadores que representem a distribuição mais recente dos dados, sendo principalmente eficiente em ambientes não estacionários. Dentro dessa estratégia três métodos podem ser empregados: reuso dos dados, filtragem e o uso de lotes de dados para formar novos classificadores [37].

Técnicas de reuso consistem em reutilizar instâncias de treino com objetivo de produzir modelos com distribuições distintas. Entre as contribuições, dois trabalhos propostos por Oza [49] se destacam: *online bagging* e *online boosting*. *Bagging* gera múltiplas versões de um conjunto de treino e utiliza esses conjuntos para obter um classificador agregado. Basicamente, a diversidade de *bagging* é obtida através de replicações de um conjunto de treinamento usando amostragem com reposição. Cada novo modelo gerado deve ter o mesmo tamanho do conjunto de treino atual. O método de amostragem permite que algumas instâncias não apareçam em certos modelos, enquanto outras apareçam mais de uma vez. *Boosting* tem como objetivo melhorar um conjunto de classificadores fracos para gerar um classificador forte. A ideia do algoritmo consiste em associar pesos a cada instância, refletindo sua importância. Ao longo das iterações, o ajuste dos pesos faz com que cada modelo aprenda sobre instâncias diferentes criando classificadores distintos. Em *boosting*, a cada iteração é gerado um modelo, e esse classificador é treinado a partir da distribuição gerada pelos pesos atuais. Após isso, com base no desempenho desse classificador, os pesos são atualizados e na próxima iteração um novo classificador é gerado com os pesos atualizados.

Filtragem é uma técnica que seleciona um conjunto especifico de instâncias para formar novos conjuntos de treino ao longo do fluxo. Segundo Breiman [13], uma forma eficiente de filtragem é gerar consecutivos classificadores utilizando um sistema chamado *Pasting small votes*. Esse sistema gera um modelo inicial utilizando todas as instâncias

disponíveis. Ao longo do tempo, novos modelos são criados a partir de dois critérios. Caso o comitê classifique incorretamente uma instância **x**, ela é selecionada para fazer parte do modelo. Por outro lado, se o comitê acertar a predição, a instância é adicionada com base em uma estimativa de erro do último classificador inserido no comitê.

Considerar o fluxo de dados em lotes ou blocos de dados é uma estratégia bastante explorada na literatura [65, 33, 59, 66, 48]. Nesse método se assume que os dados chegam em blocos de tamanho normalmente fixo. Quando um novo bloco está preenchido, o modelo é atualizado gerando um novo classificador. O modo de atualização pode ser incremental ou em *batch*, ou seja, vários modelos construídos em *batch* ou vários modelos que incorporam novos dados ao longo do tempo. Normalmente nesse tipo de estratégia utiliza algum mecanismo de esquecimento, visando manter os classificadores mais relevantes e excluir os mais antigos e/ou irrelevantes para classificação.

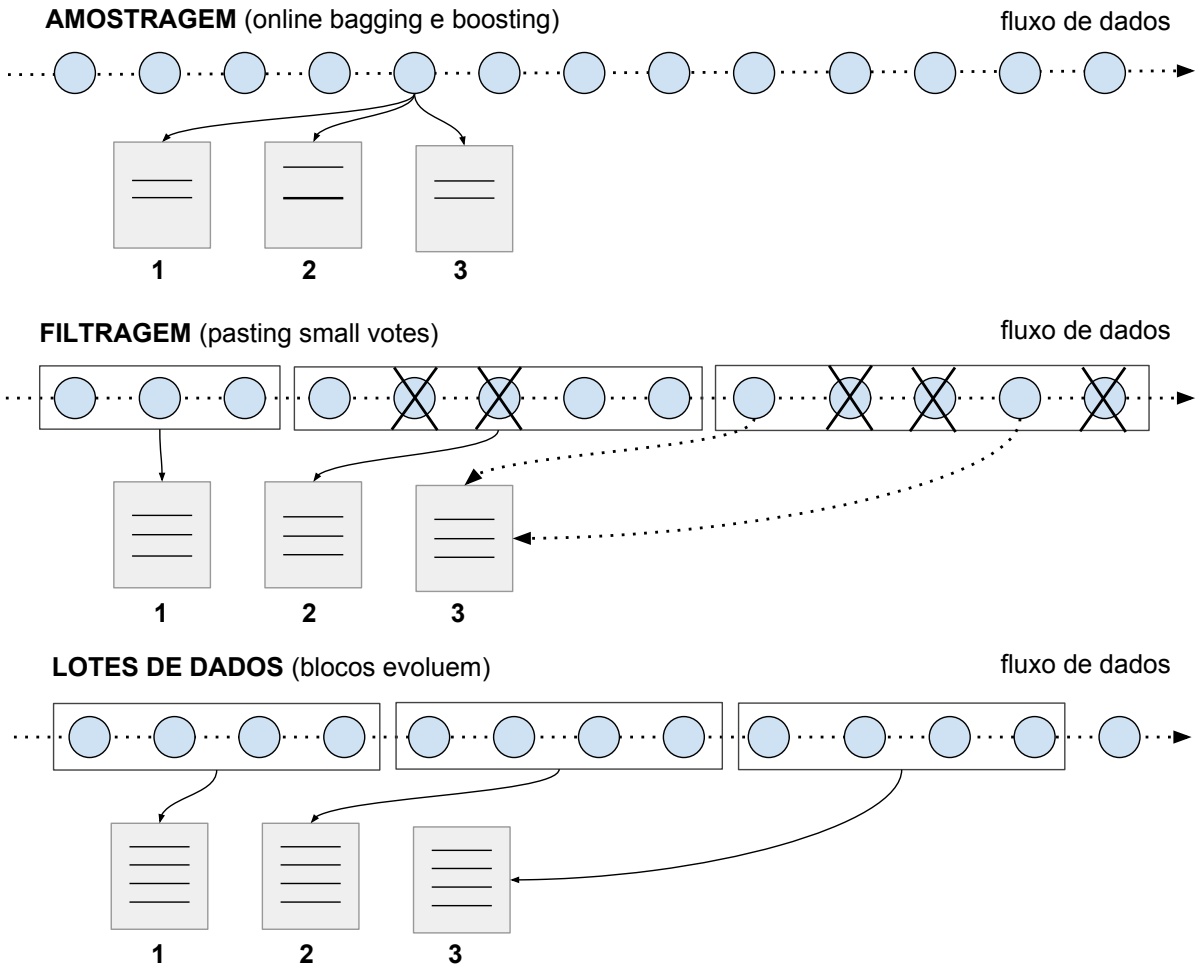

Figura 3.4 – Estratégias para o desenvolvimento de comitês de classificadores.

(c) Mudanças Estruturais do Comitê. Considerando a natural ocorrência de um *concept drift*, mudanças estruturais estão relacionadas a inclusão e exclusão dos classificadores, sendo tarefas necessárias para manter o comitê atualizado ao longo do tempo. A

estratégia mais simples consideram trocar o componente mais velho do comitê e inserir um novo componente com dados mais recentes. Apesar de ser uma ideia válida, estratégias mais sofisticadas são empregadas na literatura [65, 33, 66, 48], já que a exclusão de classificadores mais antigos não pode ser considerado um critério totalmente justificável para atualizar o comitê.

(d) Atualização dos Atributos. A necessidade de atualizar o espaço dimensional, também chamada de *feature drift*, refere-se a um tipo especial de mudança, onde é necessário adaptar o espaço de atributos para manter a qualidade preditiva. Normalmente relacionado a problemas de mineração de texto, a atualização de atributos busca manter um subconjunto de relevante de atributos para classificação, além de tratar o problema da alta dimensionalidade dos dados.

Uma vez que o espaço de atributos inicialmente é indisponível, torna-se necessário concentrar esforços na detecção e adaptação do espaço dimensional, mantendo um espaço dinâmico de atributos. Na literatura de FCD existem algumas pesquisas que exploram esse tipo de problema [31, 66, 5, 48, 26, 69]. Entretanto, poucos trabalhos como o de Wenerstrom [66], empregam a atualização de atributos em comitês de classificação.

3.7 Avaliação de classificadores em FCD

Em tarefas de mineração em FCD, a medida mais comumente usada para avaliar o desempenho preditivo de um classificador é a acurácia *prequential* [8]. No entanto, a acurácia apenas é uma boa medida quando a distribuição entre as classes está balanceada, ou seja, quando os dados de teste possuem uma quantidade de exemplos similares para cada classe considerada no problema. Por outro lado, dados de redes sociais fazem parte de um ambiente dinâmico, tendo como forte característica a mudança na distribuição dos dados e o desbalanceamento entre as classes [8]. Dessa forma, uma medida capaz de equilibrar o desbalanceamento entre as classes e fornecer estimativas mais precisas é a Estatística Kappa, proposta por Cohen [16].

A Estatística *Kappa* é tradicionalmente utilizada para medir o grau de concordância entre duas observações, além do que seria esperado tão somente pelo acaso. Para descrevermos se há ou não concordância entre dois métodos de classificação, utilizamos *Kappa* que é baseada no número de respostas concordantes, ou seja, no número de casos cujo resultado é o mesmo entre o valores reais e o que foi predito. Esta medida de concordância assume valor máximo igual a 1, que representa total concordância ou ainda pode assumir valores próximos e até abaixo de 0, os quais indicam nenhuma concordância [16]. Kappa é calculada a partir da Equação 3.2.

$$kappa = \frac{P_0 - P_e}{1 - P_e} \tag{3.2}$$

onde P_0 corresponde a acurácia e P_e é o grau de concordância caso as classes fossem previstas ao acaso.

3.8 Considerações Finais

Este capítulo apresentou os principais fundamentos que envolvem o aprendizado em FCD. Ao longo das seções foram explorados componentes de um algoritmo adaptativo, elencando questões como *concept drift*, *feature drift*, modos de aprendizado, comitês de classificação e medidas empregadas para a avaliação preditiva em FCD.

O próximo capítulo apresenta os trabalhos relacionados a esta pesquisa, explorando pesquisas que envolvem comitês de classificadores, espaços dinâmico de atributos e *concept drift*.

4. TRABALHOS RELACIONADOS

Este capítulo tem como objetivo apresentar os trabalhos que estão relacionados aos principais tópicos desta pesquisa. Na literatura foram encontrados diferentes estudos que exploram AS em FCD, espaço dinâmico de atributos e técnicas para detecção de mudança usando comitês de classificadores. Dessa forma, esses três tópicos foram divididos em diferentes seções, uma vez que nenhum dos trabalhos encontrados aborda os três assuntos em um mesmo estudo.

4.1 Análise de Sentimentos no Twitter

Na literatura de AS, diversas pesquisas exploram o Twitter e aplicam diferentes técnicas de AM para explorar a subjetividade dos textos. Entretanto, a maioria das pesquisas consideram essa tarefa como um processo *offline* [24, 50, 52, 34], o que não condiz com a natureza dos dados.

Entre os trabalhos que exploram o *Twitter* como um FCD, Bifet e Frank [8] empregam uma estratégia *test-then-train*, que incrementalmente atualiza o modelo preditivo usando os dados mais recentes. Com base em uma janela deslizante de tamanho fixo, os autores introduzem o problema do desbalanceamento de classes, propondo o uso da medida Estatística Kappa [16]. Experimentos foram realizados utilizando duas bases de dados do Twitter e três classificadores *online*: *Multinomial Naïve Bayes* (MNB), *Stochastic Gradient Descent* (SGD) e *Hoeffding Tree* (HT). Para definir o espaço dimensional, foram utilizados os dados de treino de cada base de dados, extraindo 10 mil unigramas. Os testes foram executados através do *Framework* para aprendizado *online*, Massive Online Analysis (MOA), proposto por Bifet [11]. Entre os resultados obtidos, MNB e SGD forneceram melhores resultados preditivos e de desempenho com relação a HT.

Em outro trabalho, Bifet *et* al. [10] introduzem *MOA-Tweet Hearder*, uma extensão para o Framework MOA, composto de quatro componentes principais: um coletor de *tweets* em tempo real, um classificador incremental, um detector de *concept drift* e um mecanismo para detecção de termos relevantes. O primeiro componente é responsável pela coleta e pré-processamento dos *tweets*, convertendo dados textuais para o modelo de representação BOW. O segundo componente é o classificador HT, capaz de processar a incorporar novas instâncias ao longo do tempo. O terceiro componente é ADWIN [9], um detector de mudança que considera os valores dos atributos para manter uma janela de tamanho variável, adaptando o classificador quando uma mudança é identificada. Já o quarto componente é um detector de mudança de termos, que monitora a frequência de cada palavra ao longo do tempo e fornece ao usuário informações estatísticas de mu-

dança. Experimentos foram realizados com dados da Toyota, permitindo relacionar a crise da empresa com o sentimento do público sobre a marca.

Outros trabalhos relacionados a AS envolvem comitês de classificadores. Entretanto, essas abordagens não exploram o cenário de FCD. Whitehead *et* al. [67] propõem uma análise comparativa entre um único classificador, SVM, e quatro populares comitês de classificadores: *Bagging*, *Randon Subspace* and *Bagging Subspace*, todos utilizando 50 classificadores, e *Boosting* usando 50 interações. Embora os melhores resultados foram encontrados utilizando *Randon Subspace* e *Bagging Subspace*, uma desvantagem destes métodos é o aumento do tempo computacional para treinar o classificador.

Mais recentemente, Da Silva *et* al. [17] propuseram um comitê de classificadores formado por vários classificadores: MNB, SVM, *Random Forest* e *Logistic Regression*, combinados com uma abordagem lexical. Os experimentos foram aplicados usando quatro conjuntos de dados públicos, considerando a representação *bag-of-words* e *feature hashing*. Resultados indicam que o conjunto formado por um comitê que utiliza dados textuais, *emoticons* e uma abordagem lexical melhoram os resultados da AS.

Embora existam pesquisas que explorem a AS em FCD, considerando questões como o desbalanceamento das classes e *concept drift*, nenhum dos trabalhos relacionados explora o Twitter como um espaço dinâmico de atributos. Entre os trabalhos citados nesta seção, todos definem um espaço dimensional estático, de acordo com o conjunto de treino disponível. Baseado nisso, este trabalho explora a evolução e ocorrência de *feature drift* na AS, visando adaptar o espaço dimensional em concordância com a distribuição mais recente dos dados.

4.2 Espaço Dinâmico de Atributos

Considerado um dos primeiros trabalhos a tratar o problema de *feature drift*, Katakis *et* al. [31] propõem um *Framework* que combina dois componentes principais: um método incremental para seleção de atributos e um classificador incremental. Conforme ilustra a Figura 4.1, o primeiro componente mantém um vocabulário que adiciona as novas palavras que chegam pelo fluxo e atualiza os contadores de frequência das palavras já existentes. A partir desse vocabulário, é empregado um método de seleção de atributos que, de forma incremental, elenca os *n* atributos mais relevantes e descarta aqueles que não contribuem para a classificação. Já o segundo componente corresponde a um classificador incremental, capaz de ser atualizado utilizando os atributos selecionados pelo primeiro componente. A cada nova instância **x** não rotulada, o *Framework* atualiza o classificador, que prevê a classe de **x** utilizando o subconjunto de atributos mais recente. Experimentos empregam dados de filtragem de *spam* e categorização de notícias. O trabalho utiliza a medida de seleção χ^2 e o classificador *Naïve Bayes*, uma vez que é um

técnica de classificação incremental. Apesar de relatar bons resultados, o *Framework* possui limitações no que diz respeito à restrição estática do espaço de atributos, uma vez que não utiliza nenhuma estratégia robusta para selecionar o tamanho desse subconjunto.

```
input  : Document, DocClass, Classes, Vocabulary
output: Classifier, Vocabulary, WordStats, Evaluation
begin
    foreach Word ∈ Document do
        if Word ∉ Vocabulary then
            AddWord(Word, Vocabulary)
            foreach Class ∈ Classes do
                WordStats [Word][Class][1] ← 0
                WordStats [Word][Class][0] ← 0
    foreach Word ∈ Vocabulary do
        if Word ∈ Document then
            WordStats [Word][DocClass][1] ← WordStats [Word][DocClass][1] + 1
        else
            WordStats [Word][DocClass][0] ← WordStats [Word][DocClass][0] + 1
    foreach Word ∈ Vocabulary do
        Evaluation ← EvaluateFeature(Word, WordStats)
    Classifier ← UpdateClassifier(Document, DocClass)
end
```

Figura 4.1 – *Framework* proposto por Katakis *et* al. [31], com objetivo de manter um espaço dinâmico de atributos.

Como uma extensão do trabalho de Katakis, Wenerstrom e Giraud-Carrier [66] propõem FAE (*Feature Adaptative Ensemble*), um comitê de classificadores responsável por tratar dois casos especiais de mudança: contextual e descritiva. Contextual corresponde a situações onde o conjunto de atributos deve ser modificado ao longo do tempo. Já o descritivo ocorre quando existe uma mudança entre a relação da classe alvo e os valores dos atributos, mas sem a necessidade de alterar o espaço de atributos. FAE adapta-se às mudanças do ambiente utilizando modelos de diferentes “idades”. Ao longo do tempo, os modelos são gerados, incrementalmente atualizados e o comitê adapta-se conforme os valores definidos pelos parâmetros de configuração. Para tratar mudanças contextuais, é empregada uma métrica de seleção incremental de atributos, mantendo atualizadas as estatísticas sobre os termos mais relevantes. Para atender as mudanças contextuais, um novo classificador é gerado quando o percentual de mudança nos atributos ultrapassa o limiar especificado pelo usuário. Para tratar mudanças descritivas, novos modelos são inseridos considerando o aumento do erro preditivo do comitê. A classificação de cada instância é realizada com base no peso de cada modelo: a classe que tiver maior peso é escolhida como decisão global do comitê. Para validar a proposta, são utilizados dados de filtragem de spam e *clickstream*. Assim como o trabalho de Katakis [31], FAE utiliza χ^2

como método de seleção de atributos e o classificador *Naïve Bayes*. De modo geral, FAE apresenta melhores resultados do que o trabalho de Katakis. Apesar disso, uma limitação é a quantidade de parâmetros pré-estabelecidos pelo usuários, o que torna a proposta menos flexível para atuar em ambientes com propriedades distintas.

Masud *et* al. [44] exploram o problema de espaço dinâmico de atributos para auxiliar no processo de classificação e detecção de novas classes em FCD. O algoritmo é composto por três componentes principais: um comitê de classificadores e módulos para detecção de *outliers* e de novas classes. A cada nova instância que chega pelo fluxo, inicialmente é utilizado o módulo de detecção de *outlier*, verificando se a mesma é um *outlier*. Caso não seja, a instância é rotulada pelo comitê de classificadores usando uma das classes existentes. Se a instância é um *outlier*, ela é armazenada eu um *buffer*. Quando o *buffer* estiver completo, o detector de mudanças verifica se as instâncias armazenadas nele correspondem a uma nova classe. Caso isso não ocorra, as instâncias são rotuladas pelo comitê utilizando as classes existentes. A classificação através do comitê utiliza uma técnica chamada *Lossless homogenizing conversion* (*Lossless*), que realiza a união entre o espaço dimensional de cada componente do comitê de classificadores e da instância de teste mais recente a chegar pelo fluxo, preservado os atributos de ambos [45]. *Lossless* é utilizada para auxiliar no processo de detecção de novas classes, uma vez que novas classes normalmente ocorrem junto com novos termos (atributos). Para validar a proposta são utilizados dados do Twitter, Geoespaciais e de detecção de anomalia.

Mais recentemente, Nguyen *et* al. [48] propuseram um comitê de classificadores chamado HEFT-Stream (*Heterogeneous Ensemble with Feature drifT for Data Streams*). O *Framework* proposto mantém uma técnica de seleção de atributos que periodicamente seleciona um subconjunto de atributos com base em sua relevância. Para lidar com mudanças na distribuição dos dados, um novo classificador é adicionado sempre que uma mudança abrupta ocorre ou quando o espaço dimensional muda de forma significativa. Caso o comitê esteja completo, o classificador com pior desempenho preditivo é descartado. Para maximizar a diversidade, o comitê é composto de forma heterogênea, utilizando diferentes classificadores que podem ser adicionados ao longo do tempo. Experimentos realizados empregam MNB e (*Very Fast Decision Tree*) (VFDT). Diferente das abordagens que tratam *feature drift*, HEFT-Stream se preocupa apenas com a evolução do espaço dimensional, sem incrementar atributos ao longo do fluxo de processamento.

4.3 Detecção de Mudanças usando Comitê de Classificadores

Proposto por Kolter e Maloof, *Dynamic Weighted Majority* (DWM) [33] é um comitê de classificadores baseado em WMA [40]. DWM mantém o comitê atualizado incrementalmente, combinando o voto dos classificadores através de um método ponderado. Assim

como WMA, quando um componente erra uma predição, DWM reduz seu peso multiplicando o peso atual por uma constante $\beta < 1$. Se a predição global estiver incorreta, um novo classificador é adicionado ao comitê e o peso desse classificador é inicializado com $\lambda = 1$. Para remover os classificadores ao longo do tempo, DWM mantém um limiar que verifica o peso de cada classificador: caso algum esteja abaixo desse limiar, ele é removido do comitê.

Street e Kim propõem o *Streaming Ensemble Algoritghm* (SEA) [59], uma das primeiras abordagens para endereçar o problema de *concept drift* usando comitês de classificadores. SEA mantém um comitê adaptativo, que reage as mudanças na distribuição dos dados levando em conta dois fatores principais: acurácia e diversidade, importantes fatores para garantir a eficiência de um comitê de classificadores. O algoritmo processa os dados que chegam em lotes de dados de tamanho fixo. Cada lote preenchido é utilizado para construir um novo classificador. Através de uma heurística de substituição, SEA compara esse novo classificador com todos os membros do comitê, verificando se existe um membro "mais fraco" do que o o candidato para ser membro do comitê. Caso exista um membro mais fraco, o algoritmo remove o membro mais fraco para inserir em seu lugar o novo candidato. Para realizar esse processo, inicialmente o peso dos classificadores existentes são iguais. Para cada nova instância $\mathbf{x}$ que chega, P_1 é a proporção de votos de cada componente do comitê para a classe majoritária, P_2 é a proporção de votos para a segunda classe mais votada e P_c é a proporção votada para a verdadeira classe da instância P_e é a classe predita pelo comitê. Dessa forma, o nível de cada membro do comitê é atualizada da seguinte forma:

$1 - |P_1 - P_2|$, caso o comitê e C_i sejam corretos
$1 - |P_1 - P_c|$, Caso C_i esteja correto e o comitê tenha errado
$1 - |P_i - P_c|$, caso C_i tenha errado, independente da predição do comitê

Essa estratégia é chamada de "nível de qualidade", pois não considera apenas o comitê, mas também a qualidade individual de cada classificador.

4.4 Considerações Finais

Este capítulo apresentou os principais trabalhos relacionados à esta pesquisa. Através de três seções, foram exploradas pesquisas de AS em FCD, espaço dinâmico de atributos e estratégias para o desenvolvimento de comitês de classificação dinâmicos.

Motivado pelos desafios da classificação *online* e pela limitação de trabalhos que explorem a AS como um espaço dinâmico de atributos, o próximo capítulo apresenta SENTIMENTSTREAM, um comitê de classificadores baseado em lotes, que ao longo do tempo

monitora e atualiza o espaço dimensional, além de detectar potenciais mudanças na distribuição dos dados.

5. SENTIMENTSTREAM: UM COMITÊ DE CLASSIFICADORES ADAPTATIVO PARA A ANÁLISE DE SENTIMENTO DE *TWEETS*

Neste capítulo é apresentado SENTIMENTSTREAM, um comitê de classificadores proposto para melhorar a tarefa de AS em FCD. Com foco em textos do *Twitter*, SENTIMENTSTREAM trata dois potenciais desafios da classificação *online*: *concept drift* e *feature drift*. Para lidar com esses problemas, o algoritmo proposto combina dois componentes principais: um detector de *concept drift* e um detector de *feature drift*. Juntos, os dois componentes são capazes de processar *tweets* em tempo real, adaptar-se à mudanças abruptas e incrementais, monitorar e atualizar o espaço dimensional de atributos, e principalmente, manter a qualidade preditiva ao longo do tempo. As próximas seções apresentam cada um dos componentes de SENTIMENTSTREAM.

5.1 Comitê de Classificadores Dinâmico

Esta seção descreve SENTIMENTSTREAM, um comitê de classificadores dinâmico, baseado em lotes de dados e capaz de evoluir ao longo do tempo. A inicialização do algoritmo depende de alguns parâmetros, entre eles o tamanho da janela utilizada para construir cada classificador, o número mínimo e máximo de classificadores para inicializar o comitê, e um limiar de confiança γ, utilizado para determinar espaço de atributos dos primeiros classificadores do comitê. A Tabela 5.1 apresenta todos os parâmetros de inicialização.

A Figura 5.1 apresenta uma visão geral de SENTIMENTSTREAM. Seja $\mathcal{S}$ um fluxo de dados potencialmente infinito, o algoritmo é inicializado formando uma janela J_1, a partir das p primeiras instâncias de treino disponíveis. J_1 é utilizada para gerar o primeiro classificador C_1. Após isso, J_1 é esvaziada e J_{1+1} é preenchida usando novos dados. As instâncias armazenadas em J_{1+1} são classificadas por C_1, e após isso o classificador C_{1+1} é formado usando os dados de J_{1+1}. Durante a inicialização do algoritmo, novos classificadores são gerados sequencialmente, usando o modelo mais recente para prever as instâncias da janela mais recente. Essa tarefa ocorre até que o valor $_C$ seja alcançado. $_C$ indica a quantidade mínima de classificadores necessários para iniciar do comitê. Após alcançar esse valor, um classificador novo é criado apenas na ocorrência de potenciais mudanças na distribuição dos dados ou do espaço de atributos.

Para alcançar boa assertividade, duas condições fundamentais são exploradas neste trabalho: acurácia e diversidade [18]. Para aumentar a diversidade do comitê, cada novo classificador pode ser gerado utilizando diferentes técnicas de classificação *online*, tais como MNB [46], SGD e Perceptron. SENTIMENTSTREAM não é restrito às técnicas cita-

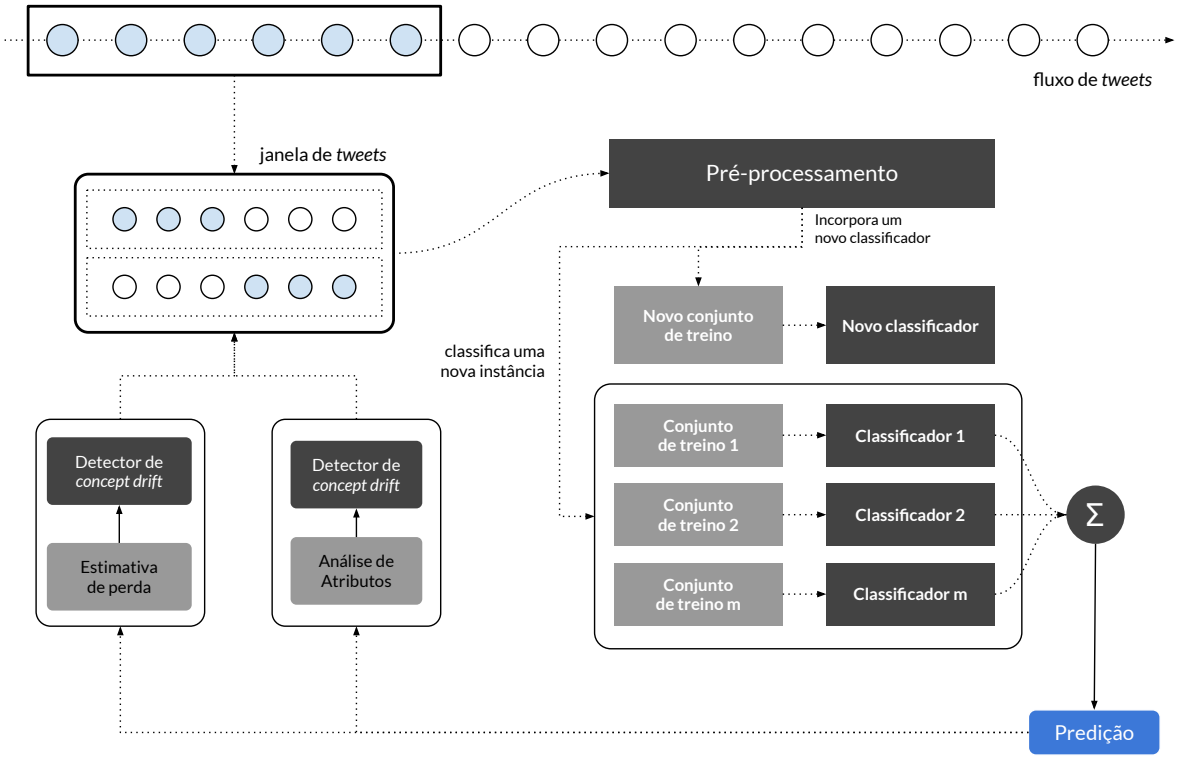

Figura 5.1 – SENTIMENTSTREAM é um comitê de classificadores baseado em lotes de dados que dinamicamente evoluem ao longo do tempo. Além dos classificadores, SENTIMENTSTREAM é composto por outros dois componentes principais: um detector de *concept drift*, que utiliza o histórico preditivo para detectar mudanças abruptas na distribuição dos dados e; um detector de *feature drift*, que monitora a ocorrência de potenciais atributos ao longo do tempo e auxilia na atualização do espaço dimensional durante o fluxo de processamento.

das; entretanto, cada algoritmo selecionado deve ter a aptidão de incorporar novos dados incrementalmente. Além disso, uma questão importante é o tempo que cada algoritmo consome para classificar e atualizar o modelo de decisão. Um exemplo desse problema ocorre com algoritmos para indução de árvore de decisão como HT. Embora seja uma técnica tradicional para classificação em FCD, segundo Bifet e Frank [8], HT não tem bom desempenho quando empregado em problemas de alta dimensionalidade e possui desempenho computacional inferior à técnicas como MNB e SGD. Dessa forma, técnicas como HT podem comprometer o processamento quando aplicadas em conjunto com algoritmos mais eficientes. Problemas relacionados ao tempo de processamento introduzem várias limitações, entre elas a criação de uma fila que mantém instâncias em memória por longos períodos de tempo ou exigência que o algoritmo descarte grandes quantidades de instâncias.

Após inicializar o comitê, cada nova janela J_l é particionada em duas sub-janelas divididas igualmente. Conforme ilustra a Figura 5.1, a primeira janela armazena a primeira metade das instâncias e a segunda armazena a outra metade. Essa estratégia é

Tabela 5.1 – Parâmetros de inicialização do algoritmo SENTIMENTSTREAM.

Parâmetro	**Nome**	**Descrição**
p	tamanho da janela	Número de instâncias utilizada para formar uma janela J, utilizada para processar os dados mais recentes e analisar o desempenho do comitê de classificação.
MIN_C	Número mínimo de classificadores	Número mínimo de classificadores (modelo de ML) necessário para inicializar o comitê e a votação ponderada.
MAX_C	Número máximo de classificadores	Número máximo de classificadores incorporados do comitê.
α	Fator de recompensa	Fator utilizado para promover um classificador quando prevê corretamente a classe de uma instância.
β	Fator de punição	Fator utilizado para "punir" um classificador que prevê incorretamente a classe de uma instância.
γ	Seleção de atributos	Índice de confiança definido para selecionar os primeiro subconjunto de atributos a partir da primeira janela de dados.
ζ	Histórico de erro preditivo	Número de janelas utilizada pelo detector de *concept drift* para calcular o histórico de erro preditivo.

utilizada em primeiro lugar para analisar e detectar mudanças abruptas na distribuição dos dados. Além disso, outro motivo é inibir falsos positivos, evitando que ruídos ou oscilações sejam consideradas como mudanças reais. Após isso, os *tweets* passam pela etapa de pré-processamento e são convertidos para a representação de BOW, que serve como entrada para o comitê de classificação. Conforme ilustra a Figura 5.2, quando uma nova instância de teste está disponível, a predição é realizada utilizando todos os componentes do comitê. Após a realizar a predição, a decisão global é realizada através de um mecanismo de voto ponderado, onde o peso de cada classificador é atualizado constantemente, de acordo com seu desempenho preditivo. Para tratar naturalmente com mudanças incrementais na distribuição dos dados, é empregada a estratégia de atualização incremental, onde os dados utilizados para teste são classificados e imediatamente usados para treinar os membros do comitê. A vantagem de utilizar uma abordagem incremental é a possibilidade de atualizar o espaço de busca dos classificadores ao longo do

tempo. Em casos onde a distribuição evolui de forma lenta, o processo incremental permite que o comitê mantenha-se atualizando em conformidade com a distribuição recente dos dados. Entretanto, quando mudanças abruptas ocorrem no ambiente, é necessário uma estratégia reativa, já que a abordagem incremental se torna inapropriada devido ao tempo demandado para atualizar o espaço de busca.

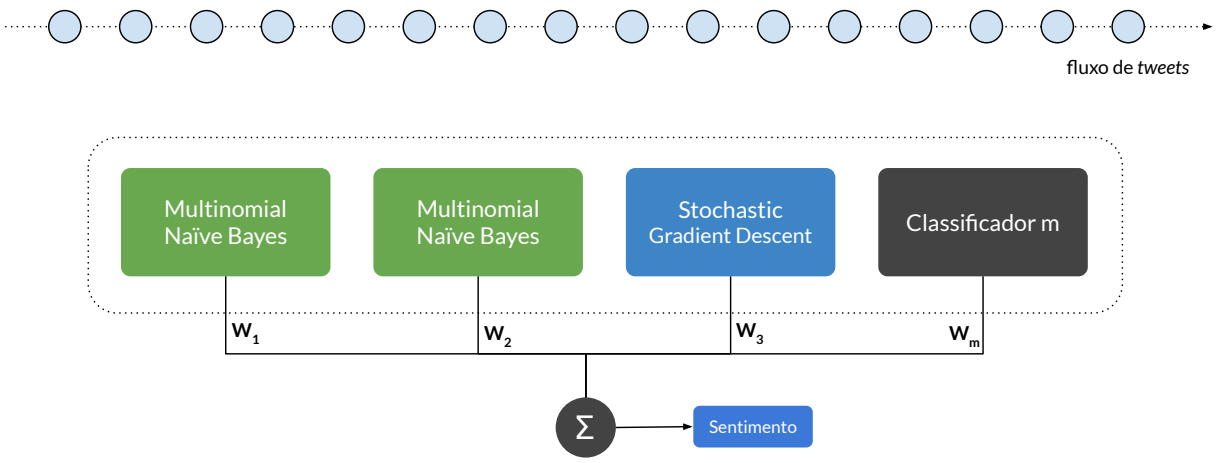

Figura 5.2 – Para cada instância de teste disponível, o algoritmo utiliza os classificadores do comitê para realizar a predição. o Sentimento final para cada instância é obtido através de uma votação ponderada, considerando o peso (relevância) de cada um dos classificadores.

Para tornar-se adaptativo, SENTIMENTSTREAM é composto por dois componentes principais. O primeiro é um detector de *concept drift*, desenvolvido para monitorar os dados mais recentes e identificar potenciais mudanças na distribuição dos dados. Para isso, o detector realiza dois tipos de análise. A primeira sobre todas as instâncias da janela mais recente e a segunda sobre cada uma das partições geradas. Inicialmente verifica-se o erro sobre a janela inteira, buscando identificar se existe um real *concept drift*. Caso o erro preditivo seja maior que um limiar pré-definido, o detector de *concept drift* dispara uma alarme para o sistema, indicando que a janela mais recente representa uma mudança na distribuição. Caso isso não ocorra, verifica-se cada uma das partições da janela. Se apenas a primeira partição representa uma mudança, ela é definida como um falso positivo, pois não prevaleceu ao longo do fluxo. Porém, se a mudança ocorrer na segunda partição, então o detector dispara outro alarme para o sistema, indicando que uma possível mudança iniciou na última partição da janela atual. Para verificar que se trata-se de uma mudança real, utiliza-se as informações da próxima janela, verificando se a primeira partição corresponde a uma mudança. Caso isso ocorra, então confirma-se uma mudança na distribuição e são utilizadas as duas partições para formar um novo classificador. Caso nenhum alarme seja disparado, então o comitê continua com os mesmos classificadores

e assume que nenhum *concept drift* ocorreu durante as janelas processadas. O segundo componente de SENTIMENTSTREAM é um detector de *feature drift*, criado com objetivo monitorar a ocorrência de potenciais mudanças espaço dimensional, ou seja, a ocorrência de novos e relevantes atributos dentro do fluxo. Uma descrição detalhada desse componente é realizada na Seção 5.5.

Caso seja detectado um *feature drift* ou *concept drift*, SENTIMENTSTREAM é atualizado, e um novo classificador é adicionado. Entretanto, esse número é limitado por $_C$, que representa o valor máximo de componentes possíveis no comitê. Quando o valor de $_C$ é atingido, então é realizado o processo de atualização por substituição. Neste caso, para adicionar um novo classificador, um classificador do comitê deve ser descartado. Para realizar essa tarefa, SENTIMENTSTREAM remove o classificador com menor peso, ou seja, aquele que mais errou predições ao longo da janela mais recente de dados. Esse processo é realizado ao longo de todo o fluxo, fazendo com que classificadores menos eficazes sejam descartados e que SENTIMENTSTREAM mantenha-se com modelos que representem de forma efetiva o espaço dimensional dos dados.

5.2 Pré-processamento de Textos Informais

Textos do Twitter introduzem uma série de desafios: entre eles estão a alta dimensionalidade, o tamanho das mensagens e o informalismo do “internetês”. Para contornar esses problemas, neste trabalho são exploradas diversas técnicas de pré-processamento, visando melhorar a representação dos dados e reduzir a dimensionalidade, evitando que atributos desnecessários sejam empregados no espaço dimensional.

Além disso, nas redes sociais é comum que textos sejam expressos usando elementos que intensificam o sentimento do usuário. Esses elementos surgem no texto em vários formatos, como: *emoticons*, *hashtags*, URLs, pontuações ou palavras alongadas. Dessa forma, neste trabalho são exploradas estratégias de limpeza e tratamento dos dados, visando obter além de palavras, elementos que intensifiquem o sentimento expresso e aumente o nível de informação sobre as mensagens. A Figura 5.3 ilustra o fluxo do tratamento dos textos adotado nesse trabalho. Seja **x** uma nova instância de teste, o processo considera desde a mensagem compartilhada pelo usuário, até sua transformação para uma representação estruturada. As estratégias selecionadas em sua maioria são conhecidas e já foram empregadas em outros trabalhos de AS. Entretanto, algumas adaptações foram realizadas visando atender os requisitos deste trabalho. Abaixo estão descritas cada uma das estratégias exploradas.

- **Filtragem:** Para eliminar ruídos nos dados, os seguintes elementos do texto foram removidos: 1) termos de consulta utilizados para coletar os *tweets*, a fim de evitar a sua influência sobre a classificação. 2) *usernames* e menções de outros usuários

no formato “@username”, e 3) caracteres especiais, tais como RT (abreviação para “ReTweet” que significa replicar algo que foi escrito) e elementos não alfanuméricos [50];

- **Tokenização:** é um processo básico em tarefas de classificação de texto. Utiliza regras e expressões regulares para dividir o texto em *tokens* (unidades básicas de uma linguagem). Esses *tokens* normalmente são palavras, pontuações e outros marcadores utilizados na linguagem [50];

- **Remoção de *stopwords*:** *Stopwords* são palavras muito comuns em um conjunto de dados e que na maioria das vezes não contribuem para a análise. Normalmente esses termos são caracterizados como pronomes, artigos e conjunções, tais como: “a”, “para”, “de”. Neste trabalho são utilizados dois dicionários de *stopwords*, um para o idioma inglês e outro para o português [50];

- **Identificação de negações:** negações são fortes indicadores de sentimento e subjetividade. Dessa forma, termos negativos como “não, nunca, jamais” são convertidos para um único atributo, chamado “_NEGACAO_” [29];

- **Identificação de pontuações:** tanto o tipo quanto a quantidade de pontuações são bons indicadores sentimento em texto subjetivos, principalmente pontos de exclamação e de interrogação. Neste trabalho, a partir de duas repetições de pontos de exclamação ou interrogação, essas pontuações são convertidas para os atributos “_EXCLAMACAO_” e “_INTERROGACAO_”, respectivamente [25];

- **Identificação de URLs:** *tweets* compartilhados pelos usuários sobre um determinado alvo podem conter características específicas, como pontuações ou URLs. Neste caso, links para páginas web, tais como sequencias de caracteres começando com “http” ou “www”, foram convertidos para um atributo chamado “_URL_”;

- **Identificação de *hashtags*** *hashtags* são formadas por símbolos no formato “#assunto”, normalmente utilizadas para relacionar o conteúdo escrito à algum tópico específico. Além de auxiliar na identificação de temas populares, *hashtags* são elementos que reforçam o sentimento de mensagens subjetivas e ajudam no processo de descoberta de conhecimento [25];

- **Identificação de *Emoticons*** em muitas mensagens, *emoticons* são utilizados pelos usuários como uma forma prática e eficiente de expressar sentimentos. Dessa forma, são considerados como elementos que favorecem a distinção de mensagens com polaridade positiva e negativa. Nesse trabalho são utilizadas duas listas contendo trinta *emoticons* de polaridade positiva e negativa, como :) :(=D =] :] =) [=, onde os *emoticons* positivos foram alterados pelo atributo “_SMILE_”, enquanto os *emoticons* negativos foram substituídos pelo atributo “_FROWN_” [25];

- **Identificação de palavras alongadas e letras maiúsculas** dois sinais fortes de subjetividade estão ligados ao uso de letras maiúsculas, como “FOME”, também conhecido como *“e-shoting”*. Assim como palavras alongadas, tal como “fooooome”. Quando uma mesma palavra é escrita usando um destes dois formatos, elas são convertidas para um atributo especial, mudando a palavra original para um *token* no formato: “_fome” [25];

- **Letras minúsculas** é comum encontrar variações na estrutura de textos compartilhados através do *Twitter*, alguns deles como: “Data Mining”, “DaTa MiNinG”, etc. Dessa forma, com intuito de normalizar os atributos e garantir a coerência no tratamento de texto, todos os caracteres são convertidos em letras minúsculas [25].

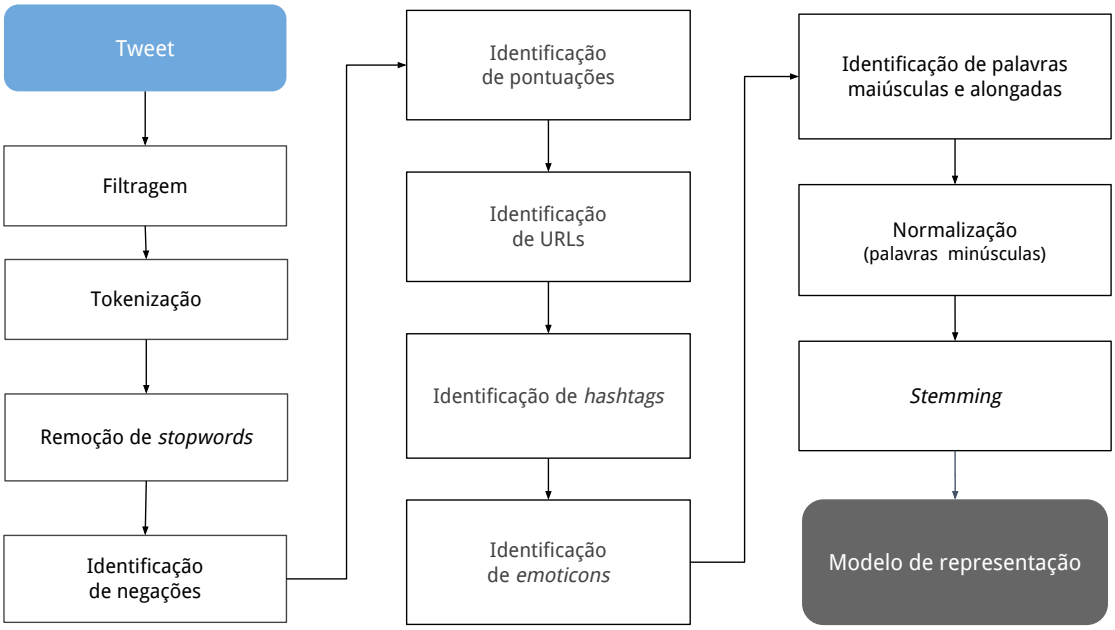

Figura 5.3 – Etapas da tarefa de pré-processamento proposto. Seja **x** uma nova instância de teste, **x** passa por todas as etapas de tratamento até ser convertida para o modelo de representação BOW.

5.3 Mecanismo de Voto

A decisão do sentimento de cada nova instância é realizada através de um método de votação ponderado, semelhante ao *Dynamic Majority Algorithm* (DMA), proposto por Littlestone e Warmuth [40]. De modo geral, o método combina a saída de cada membro do comitê de classificadores e gera um predição global Λ, usando a função *argmax*. A função *argmax* calcula a soma de todas as predições para cada classe e retorna a polaridade que obter maior votação.

Inicialmente, todos os componentes do comitê são inicializados com o mesmo peso, $w_i = 1$, sendo $i = \{1, 2, ..., m\}$, onde m é o número de classificadores que fazem parte do comitê. Para cada nova instância $(\mathbf{x}, y)$ que chega através do fluxo, a predição é realizada utilizando os pesos atuais de cada membro do comitê C_m. Diferente de DMA, o método proposto atualiza os pesos não apenas quando o comitê erra a predição, mas também quando a predição está correta. Através de uma estratégia de "recompensa", quando o comitê acerta a predição, os classificadores que também acertaram têm seu peso é multiplicado por uma constante $\alpha < 1$. Já se o comitê errar a predição, os classificadores que também erraram são "punidos", e o peso desses classificadores é multiplicado $w_i \leftarrow w_i \cdot \beta$, onde $\beta < 1$e $\alpha > \beta$. As estratégias de recompensa e punição permitem que a relevância de cada membro do comitê mantenha-se atualizado ao longo do tempo. Caso ocorra algum ruído ao longo do fluxo, classificadores com baixo desempenho no tempo t_i, podem se recuperar e de acordo com seu desempenho preditivo, tornando-se tornarem mais relevantes em $t_i + 1$. Para evitar que ao longo do tempo o voto de algum membro prevaleça substancialmente sobre as demais predições, são adotados valores pequenos para α e β.

Conforme definido, um novo modelo deve ser inserido no comitê apenas em dois casos: ocorrência de *concept drift* ou *feature drift*. Quando quaisquer dessas situações ocorrerem, o peso de todos os classificadores é reinicializado com $w_i = 1$, fazendo com que todos os classificadores iniciem o aprendizado de um novo conceito com a mesma relevância. Dessa forma, o mecanismo de votação proposto permite verificar ao longo do tempo quais os classificadores são mais relevantes para a decisão preditiva. Além disso, em comunicação com o mecanismo de detecção de mudança, é possível identificar de forma proativa os casos onde existe a necessidade de atualizar o modelo em situações onde um mudança na distribuição é gradual.

5.4 Detector de *Concept Drift*

Um *concept drift* está relacionado à mudança na distribuição que gera novos dados ao longo de um fluxo [21]. No Twitter, mudanças são comuns, indicando que o sentimento do público *online* pode mudar ao longo tempo. Entretanto, mesmo que sejam esperadas pelo sistema, tais mudanças são desconhecidas previamente, exigindo que mecanismos reativos sejam utilizados para monitorar e reagir de forma eficiente perante à um real *concept drift*. Dessa forma, neste trabalho é proposto um detector para mudanças abruptas, capaz de monitorar o ambiente de processamento e detectar potenciais mudanças na distribuição dos dados.

Seja $\mathcal{W} = \{J_1, J_2, J_3, ..., J_l\}$ a sequência das mais recentes janelas de dados observadas, e J_{l+1} uma nova janela, onde l é o total de janelas já observadas. O mecanismo

consiste em verificar se a diferença entre o desvio padrão do erro preditivo da janela mais recente é maior do que a média do desvio do erro das últimas janelas observadas. Dado um conjunto de *l* janelas, o desvio padrão é incrementalmente calculado através da Equação 5.1 [21].

$$\sigma_{J_{n+1}} = \sqrt{\frac{\sum_i^m x_i^2 - \frac{(\sum_j^l x_j)^2}{j}}{n}} \tag{5.1}$$

onde m é o número de instâncias da janela e $\sigma_{J_{l+1}}$ corresponde ao desvio padrão considerando a janela de dados mais recente. Quando o erro preditivo da janela atual for maior do que o erro preditivo das ζ janelas anteriores e $\sigma_{J_{l+1}} - \sigma_{J_l} > \delta$, em outras palavras, quando o desvio padrão atual é potencialmente maior do que o erro obtido nas janelas anteriores, então o detector irá disparar um alarme para o sistema, indicando uma mudança abrupta na distribuição (quando analisada a janela inteira) ou um alarme de uma possível mudança (quando analisada a segunda partição da janela).

Além de detectar mudanças abruptas nos dados, SENTIMENTSTREAM é capaz de naturalmente se adaptar a mudanças incrementais. Isso é possível através da estratégia de atualização incremental, uma vez que após cada instância rotulada ser utilizada como teste, ela é automaticamente incrementada em cada modelo decisão do comitê. Entretanto, apesar de atualizar cada classificador com as mesmas instâncias, SENTIMENTSTREAM tem a vantagem de manter espaços dimensionais distintos, uma vez que cada modelo mantém seu próprio espaço dimensional de acordo com a evolução do ambiente.

5.5 Detector de *Feature Drift*

Tarefas de mineração de texto em FCD se deparam com um problema em particular: *feature drift*. Esse fenômeno ocorre em ambientes dinâmicos, uma vez que o espaço dimensional não é estático, ou seja, pode sofrer mudanças ao longo do tempo. Nesse tipo de cenário, atributos relevantes podem surgir durante o fluxo e atributos conhecidos podem se tornar irrelevantes, sendo necessário adotar algoritmos que atuem sobre um espaço dinâmico de atributos.

Na literatura existem diferentes maneiras para construir espaços dinâmico de atributos [26]. Entre elas, as principais adotam um limiar fixo de termos ou um limiar fixo de relevância. Conforme ilustra a Figura 5.4, (a) na primeira estratégia são selecionados de forma empírica os N atributos mais relevantes na janela de dados mais recente. É uma forma simples e limitada, já que na maioria dos casos é impossível saber com antecedência um valor adequado para N. Além disso, essa estratégia permite facilmente que atributos irrelevantes sejam selecionados, ou em outros casos, que atributos relevantes fi-

quem fora do limiar definido. A segunda estratégia é um pouco mais eficiente. Nesse caso é definido um limiar que divide os atributos entre relevantes e irrelevantes. A vantagem dessa estratégia é a capacidade de evitar atributos irrelevantes. Entretanto, conforme mostra a Figura 5.4 (b), em casos de *drift*, o limiar fixo pode se tornar ineficiente, uma vez que atributos relevantes podem ser desconsiderados. Embora sejam estratégias intuitivas e que podem fornecer bons resultados, ambas apresentam deficiências para atuar em ambientes dinâmicos, já que a relevância de um conjunto de atributos pode ser distinta em diferentes momentos do tempo. De acordo com a Figura 5.4 (c), uma estratégia adequada é tornar esse limiar flexível, capaz de agir em concordância com a evolução do ambiente processado. Nesse sentido, este trabalho propõe um módulo especializado na evolução e ocorrência de novos atributos, capaz de monitorar mudanças no espaço dimensional e adaptar o comitê de classificadores quando potenciais mudanças ocorrem.

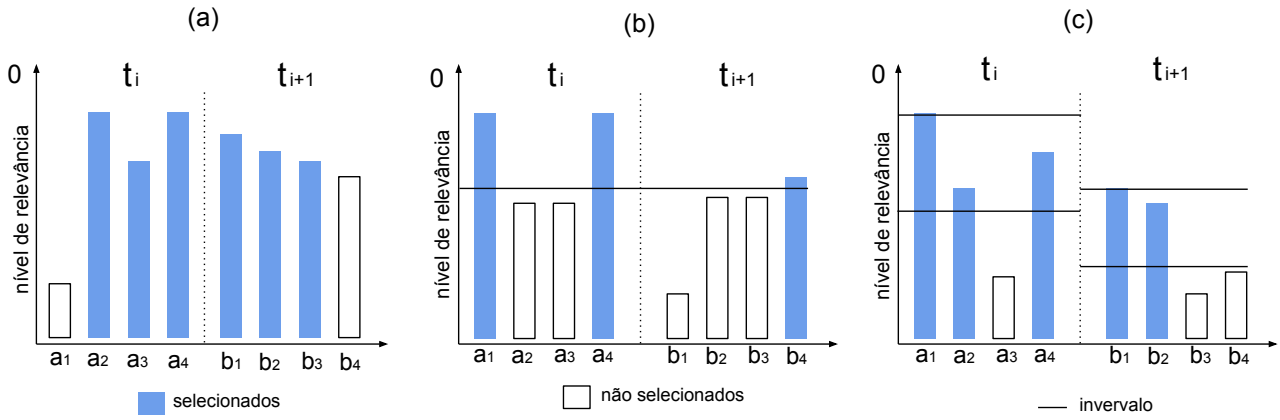

Figura 5.4 – Estratégias empregadas para construção de um espaço dinâmico de atributos. Na estratégia (a) são selecionados N atributos. Em (b) é determinado um limiar que divide os termos em relevantes e não relevantes. Já em (c) é analisado a distribuição dos dados para gerar um intervalo de atributos considerados relevantes.

Para manter um espaço dinâmico de atributos, SENTIMENTSTREAM emprega um Detector de *Feature Drift* (DFD). Trata-se de um módulo vinculado ao comitê de classificadores, que em tempo real monitora a ocorrência de novos atributos e verifica se os mesmos são relevantes para a generalização dos dados. Para identificar mudanças no espaço dimensional, DFD atua em duas etapas: (1) verificar se existe uma quantidade mínima de novos atributos dentro da janela de dados mais recente e; (2) analisar a existência de um número suficiente de atributos relevantes. Por meio dessas duas etapas é possível identificar mudanças no espaço dimensional e verificar se essa mudança é relevante o suficiente para representar um *feature drift*.

No início do processamento dois vocabulários de atributos são criados, um “recente” $V_{recente}$, que armazena estatísticas sobre os atributos da janela mais recente de dados, e um “historico” $V_{historico}$, que mantém um histórico de atributos utilizados pelos membros do comitê de classificação. Semelhante à abordagem proposta por Katakis *et*

al. [31], a cada novo *tweet* rotulado disponível no fluxo, caso alguma palavra não exista em $V_{recente}$, o termo é inserido com o valor da classe correspondente e as estatísticas de frequência do termo são inicializadas, criando um contador que inicia com o valor 1. Caso o termo esteja presente em $V_{recente}$, as estatísticas do termo são atualizadas.

Quando a janela J_l estiver preenchida, onde l representa a l-ésima janela de dados observada, DFD inicializa o processo. Conforme ilustra o Algoritmo 5.1, o detector recebe três parâmetros como entrada (linha 1): $V_{recente}$, $V_{historico}$ e IDN (Índice de Novidade), um vetor que mantém um histórico com os últimos percentuais de ocorrência de novos atributos. IDN armazena essa informação apenas quando são detectados um *feature drift* ou *concept drift*, ou seja, quando um novo classificador é inserido no comitê em virtude de uma potencial mudança no ambiente.

Para inicializar a primeira etapa de DFD, dois valores são obtidos: IDN_A e IDN_H (linhas 2-5). IDN_A representa a fração de novos atributos encontrados em J_l, ou seja, a fração de atributos presentes em $V_{recente}$ e que não estão presentes em $V_{historico}$. O valor de IDN_A é obtido por meio da Equação 5.2.

$$IDN_A = \frac{V_{recente} - V_{historico}}{len(J_l)} \tag{5.2}$$

onde $len(J_l)$ representa a quantidade de atributos de J_l e $V_{recente} - V_{historico}$ retorna a diferença entre esses dois conjuntos, ou seja, os atributos que estão em $V_{recente}$ e não estão em $V_{historico}$. Através de IDN_A, é possível identificar o percentual de novos atributos que chegam ao longo do tempo, auxiliando a detectar quando potenciais mudanças ocorrem no espaço dimensional.

Já IDN_H é um limiar histórico de "novidade", utilizado para verificar se J_l representa uma potencial mudança no espaço dimensional. De forma prática, IDN_H utiliza os valores armazenados em IDN para gerar um valor percentual. Esse valor é empregado para verificar em J_l, se existe uma quantidade mínima suficiente de novos atributos que indique um possível *feature drift*. IDN_H é obtido por meio da Equação 5.3.

$$IDN_H = IDN_{min} - \frac{IDN_{max} - IDN_{min}}{\zeta} \tag{5.3}$$

onde IDN_{min} é o menor índice de novidade em IDN e IDN_{max} é o maior índice. ζ representa a quantidade de janelas consideradas no histórico.

Após obter IDN_A e IDN_H, o algoritmo verifica se $IDN_A > IDN_H$, ou seja, se o índice de novidade atual é maior do que o índice obtido pelo histórico de novidade (linha 6). Caso o valor de IDN_A seja maior, DFD encerra a primeira etapa e inicializa a segunda, verificando se os novos atributos encontrados em J_l são também relevantes para a classificação. Entretanto, se o valor de IDN_A é menor do IDN_H, então o algoritmo descarta a possibilidade de um *feature drift* e retorna valor *False* para o sistema.

Algorithm 5.1 Detector de *Feature Drift*

IDN: Conjunto de m índices de novidade $idn_1, idn_2, ..., idn_I$
$V_{recente}$ vocabulário de atributos da janela mais recente
$V_{historico}$ vocabulário contendo atributos das últimas janelas criadas

1: **procedure** featureDrift-Detector(IDN, V_{atual}, $V_{historico}$)
2: IDN_{min} = **min**(IDN)
3: IDN_{max} = **max**(IDN)
4: $IDN_H = IDN_{min} - \frac{IDN_{max} - IDN_{min}}{\zeta}$
5: $IDN_A = \frac{V_{atual} - V_{historico}}{len(V_{historico})}$
6: **if** $IDN_A > IDN_H$ **then**
7: f = 1 - IDN_A - IDN_H
8: Med_{IDN} = **median**(IDN)
9: $Ent_{recente}$ = **entropy**($V_{recente}$)
10: *tInterval* = **trustInterval**($Ent_{recente}$, Med_{IDN}, $V_{recente}$)
11: **if** $f > tInterval$ **then**
12: *drift* = *True*
13: **else**
14: *drift* = *False*
15: **end if**
16: **else**
17: *drift* = *False*
18: **end if**
19: **return** *drift*
20: **end procedure**

Considerando que $IDN_A > IDN_H$, a segunda etapa de DFD tem como objetivo verificar se os novos atributos estão dentro de um intervalo de confiança (linhas 7-10). Nessa etapa três cálculos são necessários. O primeiro trata-se de um limiar de relevância, obtido por $f = 1 - |IDN_A - IDN_H|$. Esse valor determina o percentual mínimo de atributos de $V_{recente}$ que devem estar dentro do intervalo de confiança estabelecido. Valores de f são maiores para pequenas quantidades de atributos e menores quando existem muitos atributos novos em $V_{recente}$. O segundo valor é a mediana Med_{idn} do número de atributos utilizado no espaço dimensional de cada classificador considerado em IDN. Já o terceiro é a Entropia $Ent_{recente}$ entre os atributos de $V_{recente}$. Através do método de SA, é obtido um vetor contendo todos os atributos ordenados em ordem decrescente (mais relevante até menos relevante).

Para detectar um *feature drift*, analisa-se a seguinte situação: o limite superior é posicionado no atributo mais próximo do valor 0 em $Ent_{recente}$, ou seja, o atributo mais relevante retornado pelo método de SA. Já o limite inferior é o valor da mediana obtido por Med_{idn}. Para declarar um *feature drift*, o percentual obtido em f deve ser contemplado dentro desse intervalo, ou seja, a quantidade de novos entre o valor mais relevante e Med_{idn} deve ser pelo menos do mesmo tamanho de f. Caso isso não ocorra, *tInterval* (linha 10) retorna *False* para o sistema, indicando que a mudança foi um falso positivo.

Entretanto, se o valor de f é atendido dentro do intervalo de confiança, então o sistema retorna *True* para o sistema, indicando que um *feature drift* ocorreu. Nesse caso um novo classificador é inserido no comitê, usando o espaço dimensional de $V_recente$ e as instâncias de treino que pertencem a J_i. Em decorrência do *feature drift*, IDN é incrementado com o valor de IDN_A e com o número de atributos que retornou do intervalo de confiança. Um caso particular ocorre quando 100% dos atributos de V_{atual} são relevantes e estão dentro do limiar de confiança. Neste casos a quantidade de atributos selecionada será menor do que a mediana.

Com DFD é possível acompanhar a ocorrência de novos atributos, identificar potenciais mudanças no espaço dimensional e verificar a relevância desses atributos. Essas características permitem SENTIMENTSTREAM manter um espaço dinâmico de atributos, capaz de lidar de forma mais efetiva com o comitê de classificadores em função dessas mudanças.

5.6 Considerações Finais

Este capítulo apresentou SENTIMENTSTREAM, um comitê de classificadores dinâmico, baseado em lotes de dados e composto por dois componentes principais: um detector de *concept drift*, capaz de identificar mudanças abruptas na distribuição dos dados e; um detector de *feature drift*, que monitora a ocorrência de novos e relevantes atributos, mantendo um espaço dinâmico de atributos ao longo do fluxo de dados.

O próximo capítulo tem como objetivo apresentar os resultados de testes com SENTIMENTSTREAM, comparando-o com outras abordagens relacionadas a este trabalho.

6. RESULTADOS EXPERIMENTAIS

Esta seção tem como objetivo apresentar os resultados obtidos nesta pesquisa. Inicialmente são descritas as bases de dados utilizadas. Após isso, são destacados os resultados preliminares, destacando experimentos relacionados à tarefa de pré-processamento de dados e mudanças na distribuição do espaço dimensional - um dos problemas que motiva esta pesquisa. Por fim, são apresentados experimentos com SENTIMENTSTREAM, comparando o algoritmo proposto com abordagens relacionadas ao problema explorado.

6.1 Bases de dados

Essa seção apresenta os conjuntos de dados utilizados para testar o algoritmo proposto. São quatro bases de dados reais, com características distintas e que utilizam dados do *Twitter*. A Tabela 6.1 apresenta um sumário com as principais propriedades desses dados.

Tabela 6.1 – Características das bases de dados.

Base de dados	Idioma	Polaridades	Instâncias	Indicador de classes
Sentiment140	Inglês	POS, NEG	1,6 milhão	Emoticons
Debate BR19	Português	POS, NEU e NEG	5216	Candidato
Debate BR24	Português	POS, NEU e NEG	5745	Candidato
Eleição BR10	Português	POS, NEG	66643	Candidato

Eleição do Brasil em 2010. A Eleição Presidencial do Brasil em 2010 ocorreu entre junho e outubro. Durante esse período, a candidata Dilma Rousseff explorou um Twitter como um potencial canal para compartilhamento e monitoramento do sentimento dos eleitores. A total, a campanha alcançou mais de 500 mil seguidores e Dilma tornou-se a segunda pessoa mais citada. Ao longo do segundo turno, Universidade Federal de Minas Gerais (UFMG) coletou 466.724 *tweets* [57], onde 66.643 desses textos foram rotulados manualmente. Além disso, foram obtidos 62.082 *tweets* distintos, considerando as classes positivo e negativo. Com objetivo de analisar o sentimento e o nível de aprovação dos eleitores, todas as mensagens foram coletadas e filtradas utilizando como alvo a candidata Dilma Rousseff. A Figura 6.1 ilustra distribuição de frequência entre as classes com sentimento positivo e negativo.

Debates Eleitorais 2014 - Brasil. Durante o segundo turno da Eleição Presidencial do

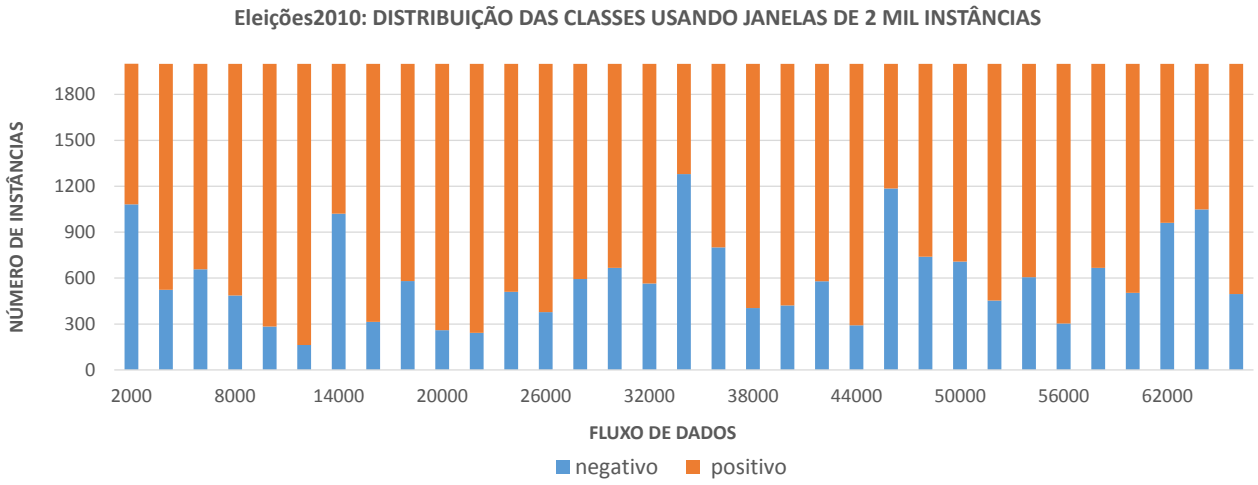

Figura 6.1 – Distribuição das classes utilizando janelas de 2 mil *tweets* na base de dados Eleição BR10.

Brasil em 2014, foram realizados quatro debates entre os candidatos, todos transmitidos pelas principais emissoras de TV do Brasil. Devido ao impacto e repercussão gerada nesses eventos, foram coletados *tweets* nos debates dos dias 19 e 24 de outubro. Como reflexo dessa popularidade, ambos também repercutiram fortemente nas redes sociais, fornecendo um significativo fluxo de opiniões *online*. Ao total foram obtidos 5216 *tweets* no debate do dia 19 e 5745 dia 24, ambos coletados entre 20h e 23:59:59 horas. Em ambos os debates, os *tweets* foram coletados utilizando como termo de consulta o candidato Aécio Neves, pois menções com o seu nome geraram uma quantidade maior de mensagem subjetivas em comparação com o outro candidato. Os termos utilizados foram "aécio" e "#aécio", e os dados foram obtidos através da API de *Streaming*[1] do *Twitter*. Os gráficos na Figura 6.2 ilustram a distribuição das classes nas duas bases de dados e as variações que ocorrem nessas distribuições.

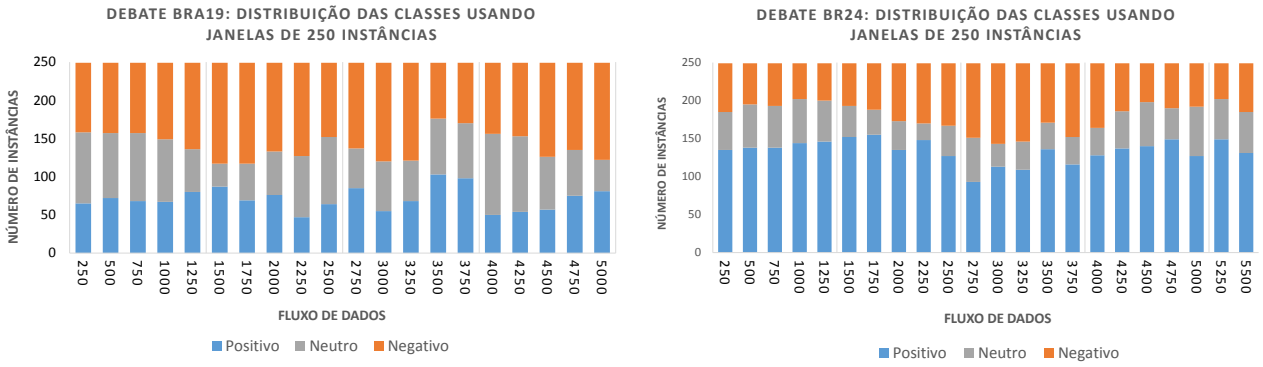

Figura 6.2 – Distribuição dos dados utilizando janelas de 250 *tweets* nas bases de dados dos Debates Eleitorais de 2014.

[1]https://dev.twitter.com/streaming/overview

Sentiment140. Proposto por Alec Go *et* al. [24], Sentiment140[2] é uma base de dados contendo conjuntos de treino e teste, obtidos a partir com a API de Streaming do *Twitter* e coletados entre 6 de abril e 25 de junho de 2009. O conjunto de treino consiste em 800 mil *tweets* positivos e 800 mil *tweets* negativos, todos escritos em inglês. A caracterização do sentimento de cada *tweet* foi mapeada a partir da presença de *emoticons* positivos, tais como :) =D =] e *emoticons* negativos, como :(:-(e :[. Neste trabalho é utilizado o conjunto de treino fornecido pelos autores, onde cada instância capturada contém a polaridade, o momento no tempo em que o *tweet* foi gerado, o nome de usuário e o próprio *tweet*.

Mesmo que o volume total do *tweets* esteja balanceado entre a quantidade de positivos e negativos, através de uma análise temporal é possível notar que a distribuição não é totalmente uniforme entre as classes. A Figura 6.3 ilustra este desbalanceamento utilizando janelas com intervalo fixo de 25 mil *tweets*, onde os mesmos são organizados em ordem cronológica com base no momento no tempo que foram gerados pelos usuários. Além disso, todos os termos de consulta utilizados nas quatro bases de dados foram removidos, impedindo sua influência no processo de classificação.

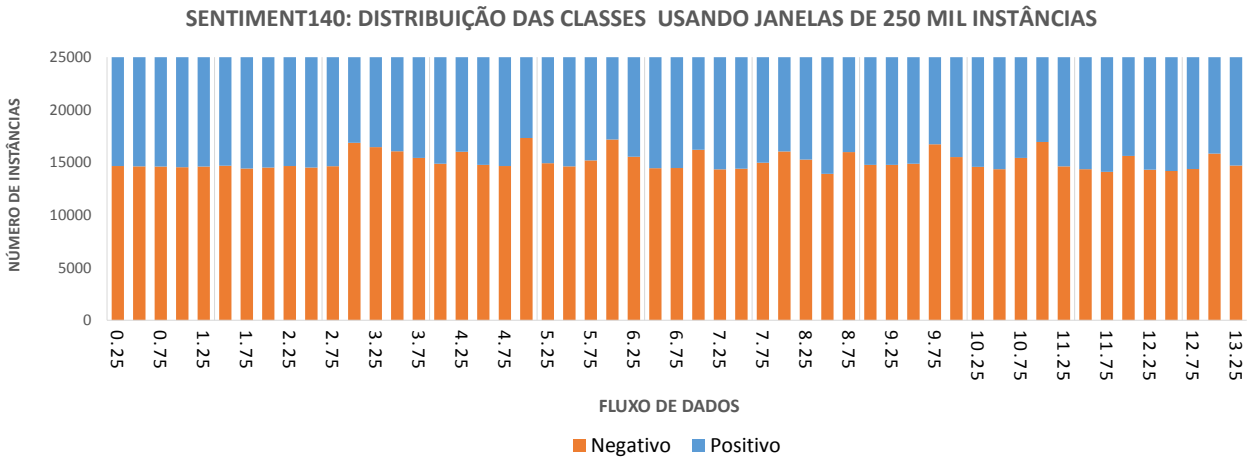

Figura 6.3 – Distribuição das classes utilizando janelas de 25 mil *tweets* em Sentiment140.

6.2 Análise Preliminar

Esta seção tem como objetivo apresentar os experimentos preliminares desta pesquisa. No que diz respeito a ambientes dinâmicos e a presença *feature drift*, duas avaliações foram realizadas. Uma delas está relacionada à mudança que ocorre na distribuição das palavras e a outra corresponde a ocorrência de novos termos ao longo do

[2] http://www.sentiment140.com/

tempo. Além disso, considerando que dados textuais não são estruturados, é realizada uma análise sobre diferentes estratégias de representação.

Os experimentos foram realizados sobre duas perspectivas: a primeira, preditiva, visando identificar quais estratégias fornecem melhores resultados de classificação. Já a segunda, relacionada ao custo computacional envolvido, uma vez que o tempo é uma questão crítica em tarefas de classificação em FCD. Dessa forma, os resultados obtidos e a discussão levantada nas próximas seções destacam um limiar entre a acurácia e a necessidade de atender aos requisitos do aprendizado *online*.

Todos os experimentos apresentados neste trabalho foram realizados utilizando um processador Intel Core i7 4770 , Memória RAM de 12 GB DDR3, NVIDIA GTX 750Ti GPU e Disco Rígido de 1TB, rodando em um sistema operacional Windows 7 64-bit. As próximas seções apresentam os experimentos realizados.

6.2.1 Mudança na distribuição das palavras

Em ambientes reais é natural que mudanças na distribuição dos dados ocorram constantemente. Em muitos casos, essas mudanças correspondem apenas a evoluções no espaço de atributos, ou seja, mudanças que não alteram a qualidade preditiva do classificador. Em outros casos, essa mudança afeta $P(y|X)$, impactando no resultado preditivo e exigindo que o modelo de decisão seja adaptado.

A Figura 6.4 ilustra um exemplo de *concept drift* do termo “acusações” ao longo do debate do dia 19 de outubro. Conforme a figura mostra, durante os dois primeiros blocos de dados, “acusações” se mantém estacionária, ou seja, com a mesma distribuição entre as classes e maior relevância para os *tweets* positivos. Entretanto, no terceiro bloco ocorre uma mudança abrupta, onde o termo deixa de ser representativo para a classe positiva e passa auxiliar a decidir o sentimento em instâncias da classe negativa. Situações como essa exigem a adaptação do modelo preditivo, uma vez que nesses casos o termo representa um novo conceito e pode afetar diretamente na decisão de $P(y|X)$.

Além da mudança na distribuição, a relevância dos atributos é um fator importante em tarefas de classificação. Além de sofrerem *concept drift*, atributos podem ao longo do tempo perder sua relevância e capacidade de decidir o sentimento de alguma classe. Dessa forma, é importante não apenas acompanhar a mudança na distribuição desses atributos, mas sim, verificar se os mesmos ainda são importantes e devem fazer parte do espaço dimensional. A Figura 6.4 apresenta essa situação no último bloco de dados, onde a frequência do termo “acusações” está igualmente distribuída entre as classes positiva e negativa. Para tratar esse tipo de problema é comum que sejam empregadas técnicas de AS. Com base na relevância dos atributos, é possível determinar quais serão utilizados e aqueles que não contribuem efetivamente para o processo de classificação.

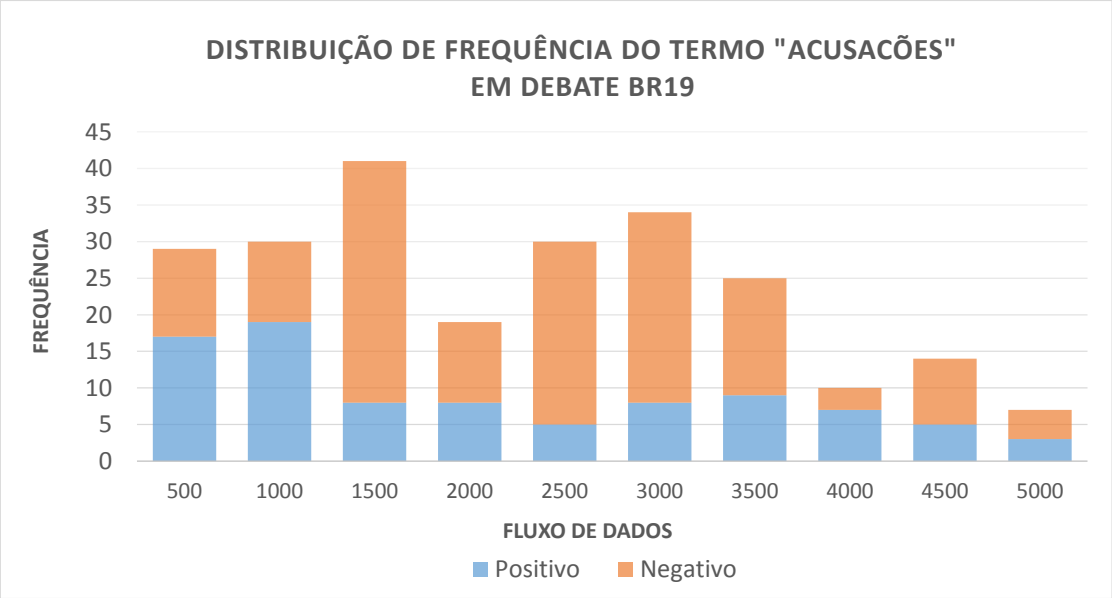

Figura 6.4 – Distribuição de frequência entre as classes positiva e negativa do termo “acusações” em Debate BR19.

6.2.2 Ocorrência de novas palavras

Além do *concept drift*, a AS enfrenta outros desafios como a alta dimensionalidade dos dados e a ausência do espaço dimensional completo no início do fluxo. Em muitas aplicações é comum que o espaço dimensional seja formado através do conjunto de treino obtido no início do processamento. Entretanto, esse tipo de abordagem é insuficiente para manipular a AS em FCD. Durante o fluxo é comum que novos termos relevantes sejam identificados, sendo necessário incorporar esses atributos no modelo preditivo. Outro problema é a alta dimensionalidade de dados. Mesmo que o espaço dimensional inicial seja limitado, em aplicações de tempo real é inviável construir um vocabulário extenso de palavras, uma vez que essa estratégia pode comprometer o desempenho do classificador e ser ineficaz para alcançar qualidade preditiva.

Para exemplificar as mudanças no espaço dimensional, a Figura 6.5 apresenta um exemplo da evolução e ocorrência de novos atributos nos dados eleitorais e *Sentiment140*. Nesse experimento, os dados são processados em blocos, onde cada bloco contém o mesmo número de instâncias. Durante o processo, são mantidos dois vocabulários, um recente, chamado de $A_{recente}$, que armazena além dos atributos, estatísticas sobre a frequência dos termos em cada uma das classes, e um vocabulário histórico, chamado $B_{historico}$, que armazena um histórico de atributos relevantes observados ao longo do fluxo. Para calcular o percentual de ocorrência de novos atributos, inicialmente é calculada a Entropia Ent_A sobre os dados de $A_{recente}$, selecionando os N atributos mais relevantes. A partir disso, é calculado INA, obtido através da equação 6.1.

$$INA = \frac{Ent_A - B_{historico}}{len(A_{recente})} \tag{6.1}$$

onde $len(A_{recente})$ representa o total de atributos em $A_{recente}$. Desta forma, INA representa o percentual de novos atributos entre os conjuntos Ent_A e $B_{historico}$, ou seja, a fração de novos atributos dentro do limiar definido que não foram vistos até o momento.

Nos debates foi utilizado $N = 300$. Dentro desse domínio é possível verificar que novos atributos relevantes surgem ao longo do fluxo. Embora esse percentual diminua, atributos relevantes são identificados em todos os blocos do fluxo. Além disso, analisando os dados do debate BR19, é possível identificar que o percentual de novos termos diminui e após isso aumenta mais de 15% a partir da segunda metade do fluxo. Esse comportamento pode ser explicado em casos onde o tópico discutido entre os usuários muda ao longo do tempo, gerando novos termos até então não vistos no espaço dimensional.

Em *Sentiment140* e Eleição BR10, devido ao tamanho das bases de dados, foram selecionados três limiares com $N = 300, 500, 1000$. O comportamento em *Sentiment140* é semelhante aos debates. Entretanto, não existe nenhuma mudança significativa na ocorrência de novos termos. Conforme era esperado, N com 1000 instâncias possui menor ocorrência ao longo do tempo e N com 300 produz maior ocorrência de novos termos. Já em Eleição BR10, exitem pelo menos dois momentos onde existe um crescimento no percentual de novos atributos, o que também indica evoluções e/ou mudanças nos tópicos discutidos.

Através dessa análise, é possível identificar a ocorrência de novos atributos relevantes ao longo do fluxo e a necessidade de adaptar os algoritmos de classificação para suportar essas mudanças. Além disso, mudanças podem ocorrer em vários momentos o tempo, sem que o sistema conheça previamente a sua natureza. Normalmente mudanças ocorrem de forma suave ou abrupta, indicando um tópico que evolui de forma incremental ou repentinamente. Dessa forma, o tratamento de espaço dinâmicos não deve apenas se restringir apenas em incorporar novos atributos, mas sim, esquecer aqueles termos que não discriminam de forma eficiente alguma das classes do problema investigado.

6.2.3 Abordagens para a representação de textos

Uma das principais etapas da AS é o pré-processamento de dados. Durante essa etapa, textos escritos em linguagem natural são convertidos para um modelo de representação estruturado, que serve como entrada para algoritmos de AM. Em AS, o modelo de representação mais comumente empregado chama-se BOW. Nesse modelo, os dados são convertidos em um vetor de atributos n-dimensional, normalmente extraídos a partir do conjunto de treino. A capacidade de BOW generalizar de forma adequada os dados

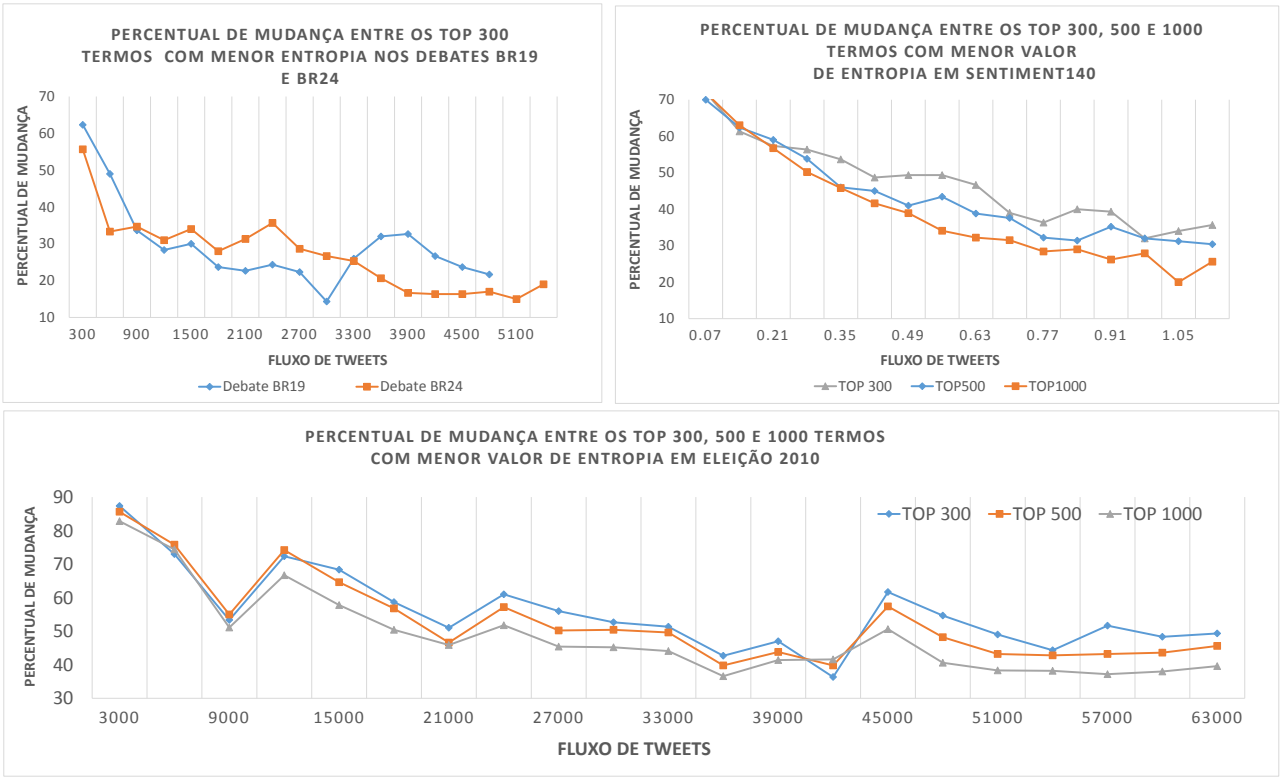

Figura 6.5 – Percentual de mudança entre os 300 termos com menor valor de Entropia nos debates BR19 e BR24.

e gerar bons resultados depende de vários fatores, entre eles: o espaço dimensional e a estratégia utilizada para atribuir pesos (valores) aos atributos. Com base nisso, esta seção realiza uma avaliação comparativa, explorando três estratégias de atribuição de pesos: binária, TF e por frequência(TF-IDF). Através dos experimentos, o objetivo é identificar quais destas estratégias contribuem no processo de classificação e se mostram mais adequadas para serem utilizadas em ambientes de FCD.

Experimentos foram realizados utilizando as quatro bases de dados propostas neste trabalho. Em Debate BR19 e Debate BR24 o modelo de decisão é gerado a partir das 300 primeiras instâncias rotuladas, usando um intervalo de amostra de 300 instâncias. Em Eleição BR10, o modelo é inicializado com 500 instâncias e o intervalo de amostragem é de 3 mil instâncias. Já em Sentiment140, o modelo de decisão contém 1000 instâncias, com um intervalo amostral de 70 mil instâncias. Valores distintos na configuração do experimento refletem o tamanho das bases de dados, visando melhorar a visualização dos resultados. Em todos os experimentos foi utilizada uma abordagem incremental, atualizando o modelo de decisão a cada nova instância rotulada. Além disso, todos os modelos foram gerados usando o classificador SGD.

A Figura 6.6 ilustra o resultado comparativo entre as estratégias binária, TF e TF-IDF. Usando os dados dos debates, TF alcançou os melhores resultados, tendo vantagem uma pequena vantagem em relação a estratégia binária. Apesar da diferença ser visual-

mente mínima, conforme apresenta a Tabela 6.2, em debate BR19, TF alcançou 52,34% de estatística *Kappa*, enquanto a abordabem binária obteve 52,15%. No debate do dia 24 o resultado foi semelhante, TF alcançou 55,11%, enquanto a binária obteve 54,27%. Em Eleição BR10, o resultado foi favorável a estratégia binária, chegando aos 63,99% de Kappa, enquanto TF obteve 63,29%. Já TF-IDF obteve os piores resultados, ficando na maioria das situações com valores de Kappa abaixo das outras abordagens exploradas. Em Sentiment140, as três abordagens alcançaram resultados semelhantes ao longo do tempo. Entretanto, diferente dos demais resultados, TF-IDF apresentou melhores resultados do que as outras abordagens, obtendo 49.16% de Kappa, enquanto TF obteve 48.57%.

Figura 6.6 – Estatística Kappa para as estratégias de representação usando abordagens binária, por frequência e TF-IDF.

Além do resultado preditivo, outra questão importante é o tempo demandado em aplicações de FCD. Apesar do bom resultado de TF-IDF em *Sentiment140*, um ponto negativo é o tempo que essa estratégia impacta no processamento. Nos experimentos realizados entre Debate BR19 e BR24, a diferença de tempo é irrelevante, ficando em torno de 2 segundos. Entretanto, em *Sentiment140* essa diferença é mais crítica, ficando em torno de 11 minutos em relação a abordagem binária, por exemplo.

Tabela 6.2 – Análise comparativa entre as três estratégias para atribuir pesos à *bag-of-words*: binária, TF e TF-IDF.

	Binária			**TF**			**TF-IDF**		
	Kp	Acc	Tp	Kp	Acc	Tp	Kp	Acc	Tp
Debate BR19	52,15	69,6	5,04	**52,34**	69,77	0,089	46,85	66,52	0,11
Debate BR24	54,27	73,24	5,63	**55,11**	73,66	0,089	49,99	71,03	0,12
Eleição BR10	**63,99**	87,37	2,81	63,29	87,21	2,79	59,58	85,91	3,61
Sentiment140	48,68	77,49	33,97	48,57	77,42	29,23	**49,16**	77,85	45,68

Em problemas de AS é comum construir o espaço dimensional utilizando apenas unigramas [64, 8, 50, 1, 25]. Entretanto, em texto curtos como *tweets*, unigramas podem limitar a capacidade de representação e a extração de informação semântica. Dessa forma, neste trabalho é realizada uma analise comparativa entre n-gramas com diferentes valores de n. Por meio dessa análise, o objetivo é verificar se bigramas, trigramas ou abordagens híbridas (usando combinações de diferentes n-gramas), auxiliam a construir um espaço dimensional mais representativo e melhoram a capacidade do algoritmo generalizar sobre os dados.

As avaliações foram realizadas com a mesma configuração do experimento anterior. Nessa análise, as seguintes abordagens foram consideradas: unigramas, bigramas, trigramas, unigramas + bigramas, unigramas + trigramas e unigramas + bigramas + trigramas.

Conforme ilustra a Figura 6.7, os resultados utilizando dados eleitorais indicam que a abordagem híbrida usando unigramas + bigramas + trigramas oferece melhores resultados que as demais. Embora o resultado seja positivo, dois aspectos limitam essa abordagem: tempo e dimensionalidade. Em relação ao tempo, abordagens híbridas utilizam um alto espaço dimensional, usando quase 5x mais atributos do que unigramas. Além disso, o tempo computacional aumenta em 4 segundos, o que pode não significar muito em uma base de dados pequena, mas pode comprometer diretamente um fluxo massivo de dados.

Em *Sentiment140*, os resultados são semelhantes. A abordagem híbrida com unigramas + bigramas + trigramas alcançou os melhores resultados e auxiliou a reduzir

o erro preditivo. Entretanto, a alta dimensionalidade e o tempo de processamento se tornaram elevados. Com um modelo preditivo usando 1000 instâncias de treino, foram utilizados 22633 atributos e o tempo de processamento foi em torno de 140 minutos, enquanto com unigramas + bigramas foi de 12227, com tempo de 104 minutos. Em ambos os casos é possível identificar a efetividade de utilizar uma abordagem híbrida, utilizando diferentes n-gramas para auxiliar a discriminar os dados. Porém os benefícios preditivos são limitados pelo custo computacional envolvido.

Já em Eleição BR10, bigramas e trigramas fornecem os piores resultados preditivos, enquanto a abordagem híbrida usando unigramas + bigramas fornece os melhores resultados, com 54.24% de Kappa, enquanto unigramas + bigramas + trigramas alcançou 54.18%, ficando em segundo lugar. No experimento em Eleição 2010 o tempo não é um fator tão crítico. Entretanto, o número de atributos gerado por unigramas + bigramas é 3x maior do que o uso de unigramas, o que não é tão impactante em um processo utilizando um único classificador, porém, pode se tornar um fator crítico quando empregado em comitês de classificação.

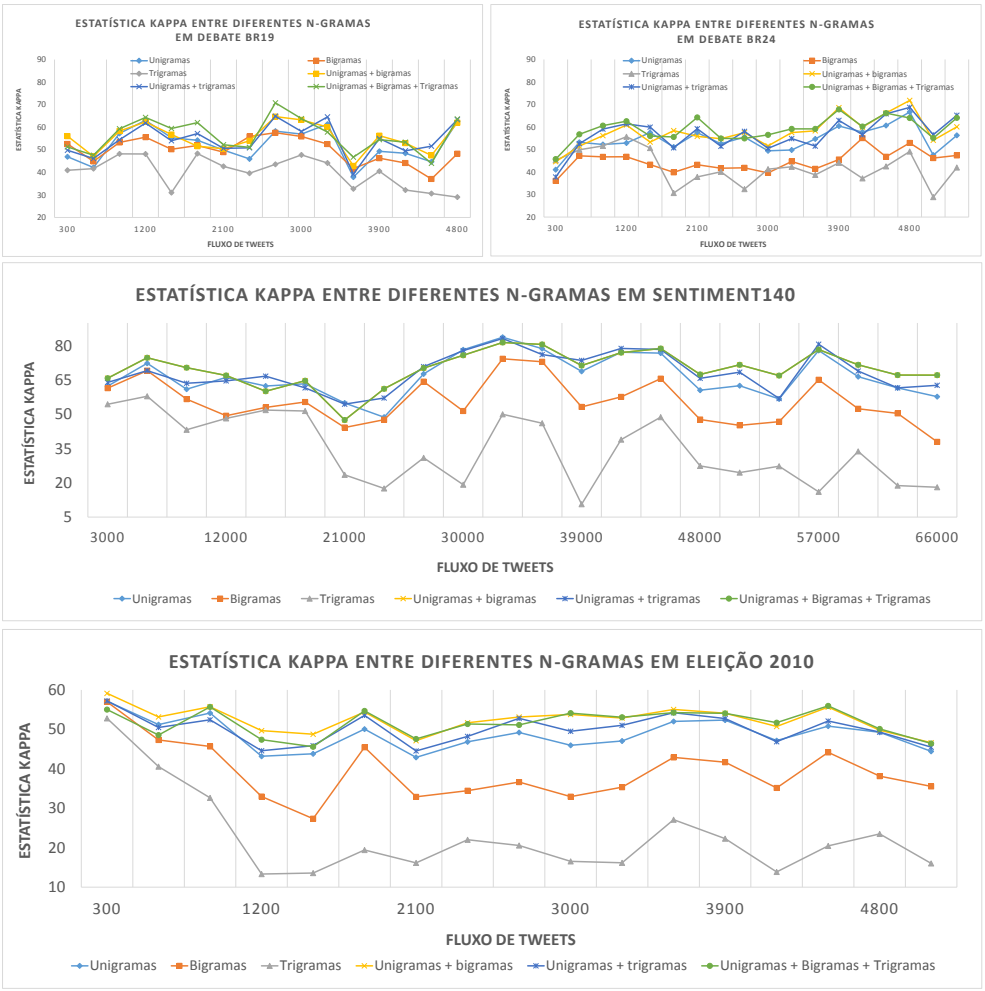

Figura 6.7 – Estatística Kappa entre diferentes n-gramas em Sentiment140 e Debates BR19 e BR24.

6.3 Experimentos com SENTIMENTSTREAM

Nessa seção são apresentados os resultados experimentais de SENTIMENTSTREAM. Para realizar os experimentos foram utilizadas quatro bases de dados reais do Twitter, três relacionadas à dados das eleições do Brasil e uma chamada *Sentiment140*, obtida através de *emoticons* positivos e negativos. Para avaliar o erro preditivo dos algoritmos, foi utilizada a medida estatística *Kappa*, empregada neste trabalho devido ao desbalanceamento entre as classes.

Para testar SENTIMENTSTREAM, foi realizada uma análise comparativa entre duas abordagens: INC, uma abordagem incremental com espaço dimensional estático e; INC.DFS, que emprega espaço dinâmico de atributos em uma abordagem de processamento incremental. A seguir são descritos os funcionamentos de INC e INC.DFS, além das configurações utilizadas em cada experimento.

INC: Nessa abordagem é utilizado um classificador incremental que atualiza o modelo preditivo utilizando as instâncias de treino mais recentes no fluxo. INC mantém o espaço dimensional estático, gerado a partir do conjunto de treino obtido inicialmente. Nos experimentos com dados dos debates foi utilizado um conjunto de treino de 300 instâncias e intervalo de amostra de 300 instâncias. Em Eleição BR10 e *Sentiment140*, o conjunto de treino foi gerado a partir das 1000 instâncias iniciais, com intervalo de 3 mil instâncias e 70 mil instâncias, respectivamente. INC foi executado utilizando o classificador SGD.

INC.DFS: Esse algoritmo é uma implementação do trabalho de Katakis *et* al. [31] Trata-se de um *Framework* que combina dois componentes: um método incremental para seleção de atributos e um classificador incremental. O primeiro componente mantém um vocabulário que incrementa as novas palavras que chegam pelo fluxo e atualiza os contadores de frequência das palavras já existentes. A partir desse vocabulário, é empregado um método de seleção de atributos que de forma incremental elenca os N atributos mais relevantes e descarta aqueles que não contribuem para a classificação. Já o segundo componente corresponde a um classificador INCREMENTAL, capaz de ser atualizado utilizando os atributos selecionados pelo primeiro componente.

A inicialização de INC.DFS é igual à de INC. Entretanto, o espaço dinâmico de atributos é atualizado em diferentes intervalos. Debate BR19 e BR24 atualizam o espaço dimensional em intervalos de 300 instâncias. Em eleição BR10, o intervalo é de 5 mil instâncias. Já em Sentiment140, esse intervalo é de 30 mil instâncias. A classificação foi realizada utilizando o classificador MNB, usando a Entropia como método de seleção de atributos. O valor de N varia de acordo com a base de dados. Nos debates o limiar é de mil atributos. Em Eleição BR10 e *Sentiment140*, o valor é de 2 mil atributos.

SENTIMENTSTREAM: O algoritmo proposto neste trabalho precisa que alguns parâmetros iniciais sejam definidos. Um deles é o tamanho das janelas de dados. Nas bases dos

debates $p = 300$. Em Eleição BR10 e *Sentiment140* $p = 1000$. O comitê é inicializado após 3 classificadores serem formados. Além disso, SENTIMENTSTREAM limita o comitê a 10 classificadores. Para construir o espaço dimensional, dos primeiros classificadores, inicialmente é utilizado um limiar de Entropia onde $\gamma = 0.8$. Após o valor de MIN ser atendido, o comitê é inicializado e são empregados os dois detectores: DCD e DFD. DCD emprega um limiar de confiança $\delta = 0.3$. Experimentos com SENTIMENTSTREAM foram realizados utilizando três técnicas de classificação: Perceptron, SGD e MNB, usando um modelo BOW híbrido, construindo o espaço dimensional com unigramas + bigramas.

Eleição BR10 contém *tweets* referente a candidata Dilma na sua primeira campanha à presidência. Conforme ilustra a Figura 6.8, SENTIMENTSTREAM apresenta melhores resultados do que as demais abordagens, mesmo tendo oscilações no desempenho preditivo ao longo do fluxo. Uma característica particular dessa base de dados é o número restrito de classificadores utilizados, embora MAX seja 10, apenas 8 classificadores são utilizados ao longo do fluxo. Esse comportamento indica que ocorrem poucas mudanças na distribuição dos dados e poucas ocorrências de potenciais mudanças na dimensionalidade. INC e INC.DFS se mostraram bastante vulneráveis na ocorrência de ruídos, tendo diversas quedas abruptas no desempenho preditivo.

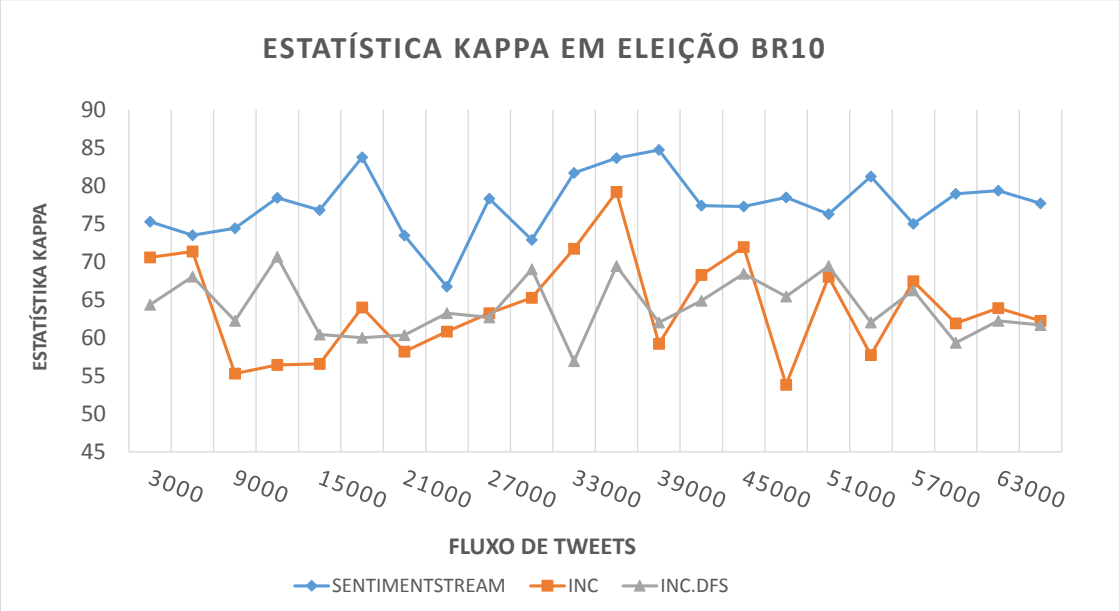

Figura 6.8 – Análise comparativa utilizando estatística Kappa em *Sentiment140*.

A Figura 6.9 ilustra o desempenho preditivo de SENTIMENTSTREAM em Debate BR19. Com o resultado obtido é possível verificar o algoritmo proposto nesse trabalho alcança melhores resultados do que as abordagem INC e INC.DSF. Além disso, SENTIMENTSTREAM é capaz de ser eficaz na ocorrência de *concept drift*, mantendo seu desempenho melhor durante mudanças na distribuição dos dados e ruídos. Na ocorrência de *concept drift*, a abordagem INC sofre ligeiras quedas do desempenho preditivo ao longo do processamento, obtendo o pior desempenho entre os três algoritmo comparados. INC.DSF

mantém melhor desempenho do que INC. Entretanto, essa abordagem sofre com as mudanças constantes na distribuição das classes. Apesar de INC.DSF manter um espaço dinâmico de atributos, a constante mudança do espaço dimensional utilizando um limiar empírico prejudica ao desempenho do algoritmo.

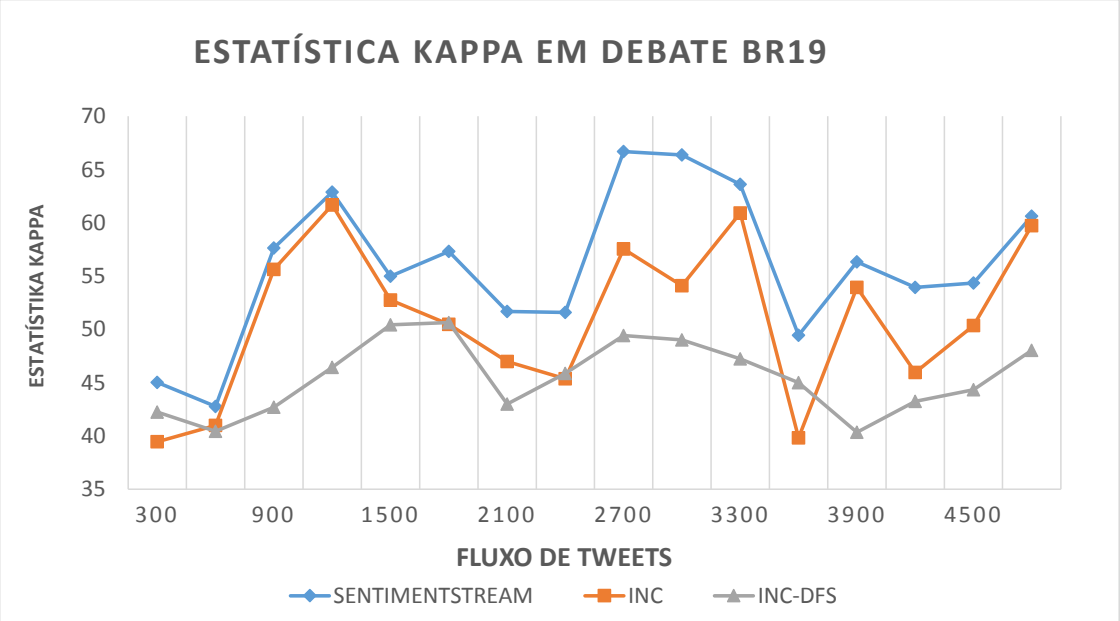

Figura 6.9 – Análise comparativa utilizando estatística Kappa em Debate BR19.

Em debate BR24, conforme ilustra a Figura 6.10, SENTIMENTSTREAM apresenta resultado semelhante à Debate BR19, obtendo melhores valores de Kappa do que as outras abordagens. A abordagem INC mais uma vez obteve desempenho inferior as demais abordagens e se mostrou bastante vulnerável na presença de *concept drift*. Já INC.DSF manteve desempenho abaixo de SENTIMENTSTREAM. INC.DSF se mostrou mais eficiente do que a abordagem INC. Porém, na ocorrência de *concept drift*, ambas as abordagens se mostraram inferiores. Uma razão para isso é a ausência de um detector de mudança. Embora o modelo preditivo seja incrementalmente atualizado, na ocorrência de mudanças abruptas, o algoritmo demora para se adaptar. Uma limitação em INC é o fato de manter um espaço dimensional fixo, criado no início do fluxo, usando os dados de treino disponíveis.

Uma característica dos dados eleitorais é a presença de ruídos ao longo do fluxo, ou seja, mudanças constantes na distribuição dos dados. Além da frequência, essas mudanças são repentinas, fazendo com que o algoritmo fique suscetível à falsos positivos, ou seja, ruídos que são tratados como mudanças abruptas ao longo do fluxo. Para evitar esse problema em SENTIMENTSTREAM, toda vez que uma mudança ocorre, o detectores de *concept drift* ou *feature drift* ficam em estado de espera até que duas novas janelas sejam observadas, sendo ativados a partir da terceira janela. Esse tipo de estratégia evita

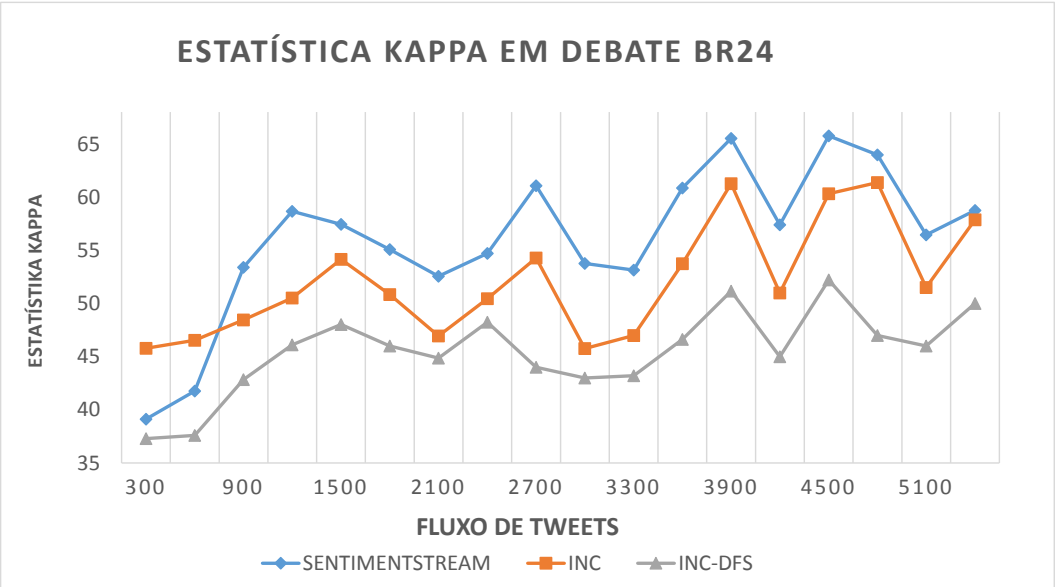

Figura 6.10 – Análise comparativa utilizando estatística Kappa em Debate BR24.

que essas constantes mudanças prejudiquem a eficácia do algoritmo e auxilia para que verdadeiras mudanças na distribuição sejam detectadas.

Em *Sentiment140*, SENTIMENTSTREAM alcançou novamente bons resultados. Conforme ilustra Figura 6.11, SENTIMENTSTREAM alcançou os melhores valores de Kappa ao longo do processamento. Além disso, o comitê se mostrou robusto para mudanças na distribuição, sendo menos afetado do que as outras abordagens. Diferente disso, as abordagens INC e INC.DSF têm resultados que oscilam ao longo do tempo, ficando abaixo de SENTIMENTSTREAM ao longo do processamento.

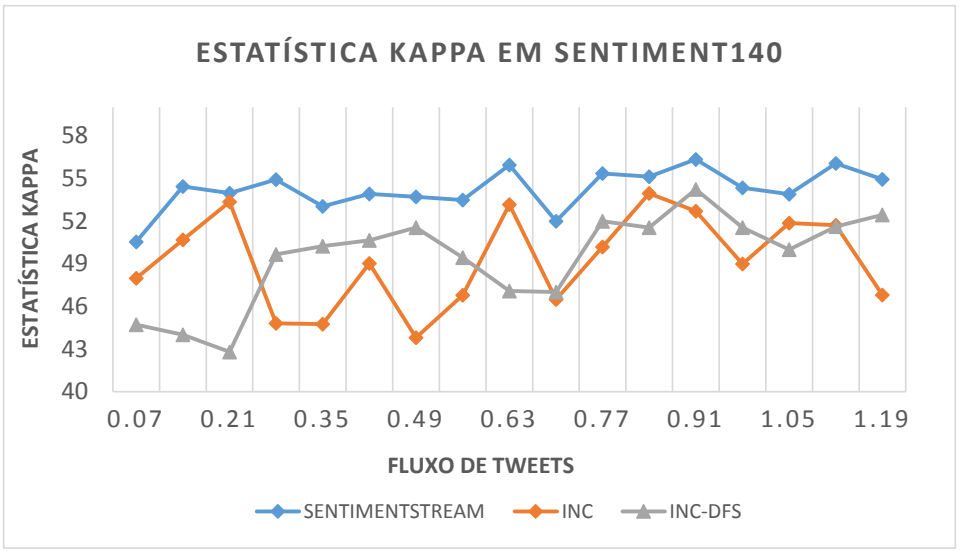

Figura 6.11 – Análise comparativa utilizando estatística Kappa em *Sentiment140*.

Além da capacidade preditiva, a Figura 6.12 apresenta uma avaliação comparativa do tempo de execução entre SENTIMENTSTREAM, INC e INC.DFS. Nos debates eleitorais, SENTIMENTSTREAM se mostra inferior as demais abordagens, demandando mais tempo do que INC e INC.DFS. Essa desempenho se deve à alguns fatores. Um deles é a presença de do comitê de classificação, que prevê cada nova instância no fluxo usando vários classificadores. Além disso, SENTIMENTSTREAM é composto por dois detectores de mudança, que monitoram em tempo real as mudanças que ocorrem no ambiente. Através dos gráficos dos dados eleitorais, é possível notar que o tempo aumenta progressivamente em SENTIMENTSTREAM. Esse aumento ocorre em função do aumento de classificadores no comitê, estabilizando quando o comitê alcança seu nível máximo de membros. INC tem o menores resultados de tempo, tendo desempenho praticamente linear ao longo do tempo. Já INC.DFS possui pequenas oscilações, todas elas em função do espaço dinâmico de atributos empregado neste algoritmo.

Figura 6.12 – Análise comparativa do tempo de execução entre SENTIMENTSTREAM e as abordagens INC e INC.DFS.

Em Sentiment140, o tempo consumido para processar os dados é constante e consideravelmente maior do que as outras abordagens. A linearidade de SENTIMENTSTREAM se deve ao limite no número de membros do comitê, permitindo que ele mantenha custos semelhantes pra processar cada janela ao longo do fluxo. INC e INC.DFS possuem resultados lineares também. Entretanto, INC demanda menos tempo, já que mantém um único classificador e apenas incrementa instâncias no modelo ao longo do tempo.

Os resultados de SENTIMENTSTREAM são motivados por vários fatores. Um desses fatores é a capacidade do comitê evoluir ao longo do tempo, formando um comitê de classificadores compostos por espaços dimensionais distintos e com atributos relevantes. Além disso, SENTIMENTSTREAM constrói um comitê heterogêneo, formado por diferentes técnicas de aprendizado e modelos preditivos que são incrementados a cada nova instância de treino disponível através do fluxo. Para apoiar o fluxo de classificação, os detectores de *feature drift* e *concept drift* auxiliam a detectar mudanças abruptas na distribuição dos dados e potenciais mudanças no espaço dimensional. Dessa forma, SENTIMENTSTREAM é capaz de manter a qualidade preditiva, reagindo na ocorrência de mudanças no ambiente. Com os resultados obtidos, é possível verificar que SENTIMENTSTREAM contribui de forma significativa para AS em FCD, fornecendo resultados positivos para esse tipo de problema.

6.4 Considerações Finais

Este capítulo apresentou os resultados dos experimentos para testar SENTIMENTSTREAM. Inicialmente foi realizada uma análise preliminar, considerando mudanças no espaço dimensional e a ocorrência de *concept drift*. Após isso foram apresentados os resultados referentes a SENTIMENTSTREAM, comparando o algoritmo proposto com outras duas abordagens.

Resultados indicam que SENTIMENTSTREAM é capaz de auxiliar a melhorar a AS em FCD. No próximo capítulo são apresentadas as conclusões e trabalhos futuros.

7. CONCLUSÕES E TRABALHOS FUTUROS

Devido a grande quantidade de conteúdo opinativo compartilhada diariamente, o *Twitter* tornou-se uma ferramenta de comunicação importante e valiosa fonte de dados em tarefas de AS. Embora muitos trabalhos considerem essa tarefa como um processo *offline*, o Twitter caracteriza-se como uma aplicação em FCD, onde os *tweets* chegam em tempo real, em alta velocidade e sobre uma distribuição que pode mudar ao longo do tempo.

Tendo em vista as características e os desafios da AS em FCD, este trabalho propõe SENTIMENTSTREAM, um comitê dinâmico de classificadores, baseado em lotes de dados e desenvolvido para classificar o sentimento de *tweets* em FCD. SENTIMENTSTREAM é composto por dois componentes principais: (i) um detector de *concept drift*, capaz de detectar e reagir de forma eficiente a mudanças abruptas na distribuição dos dados; e (ii) um detector de *feature drift*, que utiliza uma estratégia automática para monitorar e identificar potenciais mudanças no espaço dimensional. Através de SENTIMENTSTREAM, o objetivo é aprimorar a tarefa de AS, considerando os desafios relacionados a informalidade dos textos de redes sociais e as mudanças no ambiente de processamento.

Os experimentos deste trabalho foram realizados utilizando quatro bases de dados reais do Twitter. Inicialmente foram conduzidos experimentos preliminares, explorando diferentes estratégias de pré-processamento sobre dois aspectos principais: tempo e qualidade preditiva. Além disso, foi realizada uma avaliação sobre a evolução dos atributos dentro de cada base de dados. Resultados dessa avaliação indicam que existe a necessidade de manter um espaço dimensional dinâmico, já que novos atributos relevantes surgem ao longo do fluxo de dados e atributos conhecidos podem se tornar irrelevantes.

Para testar SENTIMENTSTREAM, foram realizados experimentos comparativos com duas abordagem que atuam em FCD. A primeira, chamada INC, utilizando uma abordagem de aprendizado incremental, usando um único algoritmo e um espaço dimensional estático. A segunda, chamada de INC.DFS, mantém um espaço dinâmico de atributos, armazenando um vocabulário de atributos que seleciona os N termos mais relevantes e retreina o classificador periodicamente.

Resultados demonstram que SENTIMENTSTREAM contribui de forma significativa para AS em FCD, fornecendo resultados positivos para esse tipo de problema. Enquanto as abordagens INC e INC.DFS oscilam na presença de *concept drift*, SENTIMENTSTREAM se mantém com menor erro preditivo ao longo do tempo. Os resultados de SENTIMENTSTREAM são motivados por vários fatores. Um deles é a capacidade do comitê evoluir ao longo do tempo, formando um comitê de classificadores compostos por espaços dimensionais distintos e com atributos relevantes. Além disso, SENTIMENTSTREAM constrói um comitê heterogêneo, formado por diferentes técnicas de aprendizado e modelos predi-

tivos que são incrementados a cada nova instância de treino disponível. Para apoiar o fluxo de classificação, os detectores de *concept drift* e *feature drift* atuam de forma complementar, verificando a ocorrência de mudanças abruptas na distribuição dos dados e a ocorrência de novos atributos relevantes.

Embora os resultados sejam positivos, limitações pontuais são encontradas neste trabalho. Uma delas é o tempo demandado para processar os dados. Mesmo que comitês sejam mais precisos do que um único classificador, o tempo computacional é um fator crítico nesse processo. Em trabalhos futuros, o objetivo dessa pesquisa é explorar cenários com poucos dados rotulados, explorando técnicas de aprendizado semisupervisionado para manter boa acurácia nesses casos. Além disso, para reduzir o impacto do tempo computacional, o objetivo é explorar técnicas de computação distribuída, com objetivo de explorar de forma mais eficiente os recursos computacionais disponíveis.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Agarwal, A.; Xie, B.; Vovsha, I.; Rambow, O.; Passonneau, R. “Sentiment analysis of twitter data”. In: Workshop on Languages in Social Media, 2011, pp. 30–38.

[2] Aggarwal, C.; Zhai, C. “Mining text data”. Springer, 2012, 524p.

[3] Agichtein, E.; Castillo, C.; Donato, D.; Gionis, A.; Mishne, G. “Finding high-quality content in social media”. In: International Conference on Web Search and Data Mining, 2008, pp. 183–194.

[4] Babcock, B.; Babu, S.; Datar, M.; Motwani, R.; Widom, J. “Models and issues in data stream systems”. In: ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, 2002, pp. 1–16.

[5] Barddal, J. P.; Gomes, H. M.; Enembreck, F. “Analyzing the impact of feature drifts in streaming learning”. In: Neural Information Processing, 2015, pp. 21–28.

[6] Becker, K.; Tumitan, D. “Introdução à mineração de opiniões: Conceitos, aplicações e desafios”. In: Simpósio Brasileiro de Banco de Dados, 2013, pp. 27–52.

[7] Bhaskar, J.; Sruthi, K.; Nedungadi, P. “Enhanced sentiment analysis of informal textual communication in social media by considering objective words and intensifiers”. In: Recent Advances and Innovations in Engineering, 2014, pp. 1–6.

[8] Bifet, A.; Frank, E. “Sentiment knowledge discovery in twitter streaming data”. In: International Conference on Discovery Science, 2010, pp. 1–15.

[9] Bifet, A.; Gavalda, R. “Learning from time-changing data with adaptive windowing”. In: SIAM International Conference on Data Mining, 2007, pp. 443–448.

[10] Bifet, A.; Holmes, G.; Pfahringer, B.; Gavalda, R. “Detecting sentiment change in twitter streaming data”. In: Workshop on Applications of Pattern Analysis, 2011, pp. 5–11.

[11] Bifet, A.; Holmes, G.; Pfahringer, B.; Read, J.; Kranen, P.; Kremer, H.; Jansen, T.; Seidl, T. “Moa: a real-time analytics open source framework”. In: Machine Learning and Knowledge Discovery in Databases, 2011, pp. 617–620.

[12] Bollen, J.; Mao, H.; Zeng, X. “Twitter mood predicts the stock market”, *Journal of Computational Science*, vol. 2–1, 2011, pp. 1–8.

[13] Breiman, L. “Pasting small votes for classification in large databases and on-line”, *Machine learning*, vol. 36–1, 1999, pp. 85–103.

[14] Brzeziński, D. “Mining data streams with concept drifts”. (master’s thesis), Faculty of Computing Science and Management, Poznan University of Technology, Poznan, Poland, 2010, 89p.

[15] Carmona-Cejudo, J.; Baena-García, M.; del Campo-Ávila, J.; Morales-Bueno, R.; Bifet, A. “Gnusmail: Open framework for on-line email classification”. In: European Conference on Artificial Intelligence, 2011, pp. 1141–1142.

[16] Cohen, J. A. “A coefficient of agreement for nominal scales”. In: Educ Psychol Meas, 1960, pp. 37–46.

[17] da Silva, N. F.; Hruschka, E. R.; Hruschka, E. R. “Tweet sentiment analysis with classifier ensembles”, *Decision Support Systems*, vol. 66, 2014, pp. 170–179.

[18] Dietterich, T. G. “Ensemble methods in machine learning”. In: Multiple Classifier Systems, 2000, pp. 1–15.

[19] Feldman, R. “Techniques and applications for sentiment analysis”, *Communications of the ACM*, vol. 56–4, 2013, pp. 82–89.

[20] Feldman, R.; Sanger, J. “The text mining handbook: advanced approaches in analyzing unstructured data”. Cambridge University Press, 2007, 424p.

[21] Gama, J. “Knowledge Discovery from Data Streams”. Chapman & Hall/CRC, 2010, 1st ed., 244p.

[22] Gama, J.; Medas, P.; Castillo, G.; Rodrigues, P. “Learning with drift detection”. In: Advances in Artificial Intelligence, 2004, pp. 286–295.

[23] Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; Bouchachia, A. “A survey on concept drift adaptation”, *ACM computing surveys*, vol. 46–4, 2014, pp. 1–37.

[24] Go, A.; Bhayani, R.; Huang, L. “Twitter sentiment classification using distant supervision”, *CS224N Project Report, Stanford*, vol. 1, 2009, pp. 1–6.

[25] Gokulakrishnan, B.; Priyanthan, P.; Ragavan, T.; Prasath, N.; Perera, A. “Opinion mining and sentiment analysis on a twitter data stream”. In: International Conference on Advances in ICT for Emerging Regions, 2012, pp. 182–188.

[26] Gomes, J. B.; Gaber, M. M.; Sousa, P.; Menasalvas, E.; et al.. “Mining recurring concepts in a dynamic feature space”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25–1, 2014, pp. 95–110.

[27] Hilas, C. S. “Designing an expert system for fraud detection in private telecommunications networks”, *Expert Systems with applications*, vol. 36–9, 2009, pp. 11559–11569.

[28] Jansen, B. J.; Zhang, M.; Sobel, K.; Chowdury, A. “Twitter power: Tweets as electronic word of mouth”, *Journal of the American Society for Information Science and Technology*, vol. 60–11, 2009, pp. 2169–2188.

[29] Ji, X.; Chun, S. A.; Geller, J. “Monitoring public health concerns using twitter sentiment classifications”. In: IEEE International Conference on Healthcare Informatics, 2013, pp. 335–344.

[30] Katakis, I.; Tsoumakas, G.; Banos, E.; Bassiliades, N.; Vlahavas, I. “An adaptive personalized news dissemination system”, *Journal of Intelligent Information Systems*, vol. 32–2, 2009, pp. 191–212.

[31] Katakis, I.; Tsoumakas, G.; Vlahavas, I. “Dynamic feature space and incremental feature selection for the classification of textual data streams”, *Knowledge Discovery from Data Streams*, 2006, pp. 107–116.

[32] Kelly, M.; Hand, D. J.; Adams, N. M. “The impact of changing populations on classifier performance”. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 367–371.

[33] Kolter, J. Z.; Maloof, M. A. “Dynamic weighted majority: An ensemble method for drifting concepts”, *The Journal of Machine Learning Research*, vol. 8, 2007, pp. 2755–2790.

[34] Kouloumpis, E.; Wilson, T.; Moore, J. “Twitter sentiment analysis: The good the bad and the omg!” In: AAAI Conference on Web and Social Media, 2011, pp. 538–541.

[35] Kumar, P. R.; Ravi, V. “Bankruptcy prediction in banks and firms via statistical and intelligent techniques - a review”, *European Journal of Operational Research*, vol. 180–1, 2007, pp. 1–28.

[36] Kuncheva, L. “Classifier ensembles for detecting concept change in streaming data: Overview and perspectives”. In: Workshop SUEMA, 2008, pp. 5–10.

[37] Kuncheva, L. I. “Classifier ensembles for changing environments”. In: International Workshop on Multiple Classifier Systems, 2004, pp. 1–15.

[38] Largeron, C.; Moulin, C.; Géry, M. “Entropy based feature selection for text categorization”. In: ACM Symposium on Applied Computing, 2011, pp. 924–928.

[39] Lindstrom, P.; Delany, S. J.; Mac Namee, B. “Handling concept drift in a text data stream constrained by high labelling cost”. In: International FLAIRS Conference, 2010, pp. 19–21.

[40] Littlestone, N. “Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm”, *Machine learning*, vol. 2–4, 1988, pp. 285–318.

[41] Littlestone, N.; Warmuth, M. K. “The weighted majority algorithm”, *Information and Computation*, vol. 108–2, 1994, pp. 212–261.

[42] Liu, B. “Sentiment analysis and opinion mining”, *Synthesis Lectures on Human Language Technologies*, vol. 5–1, 2012, pp. 1–167.

[43] Liu, B.; et al.. “Sentiment analysis and subjectivity”, *Handbook of Natural Language Processing*, vol. 2–2010, 2010, pp. 627–666.

[44] Masud, M. M.; Chen, Q.; Khan, L.; Aggarwal, C. C.; Gao, J.; Han, J.; Srivastava, A.; Oza, N. C. “Classification and adaptive novel class detection of feature-evolving data streams”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 25–7, 2013, pp. 1484–1497.

[45] Masud, M. M.; Gao, J.; Khan, L.; Han, J.; Thuraisingham, B. “Classification and novel class detection in concept-drifting data streams under time constraints”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 23–6, 2011, pp. 859–874.

[46] McCallum, A.; Nigam, K.; et al.. “A comparison of event models for naive bayes text classification”. In: AAAI Workshop on Learning for Text Categorization, 1998, pp. 41–48.

[47] Moreira, J. M.; Soares, C.; Jorge, A. M.; de Sousa, J. F. “The effect of varying parameters and focusing on bus travel time prediction”. In: Advances in Knowledge Discovery and Data Mining, 2009, pp. 689–696.

[48] Nguyen, H.-L.; Woon, Y.-K.; Ng, W.-K.; Wan, L. “Heterogeneous ensemble for feature drifts in data streams”. In: Advances in Knowledge Discovery and Data Mining, 2012, pp. 1–12.

[49] Oza, N. C.; Russell, S. J. “Online bagging and boosting”. In: International Workshop on Artificial Intelligence and Statistics, 2001, pp. 229–236.

[50] Pak, A.; Paroubek, P. “Twitter as a corpus for sentiment analysis and opinion mining”. In: International Conference on Language Resources and Evaluation, 2010, pp. 17–23.

[51] Pang, B.; Lee, L. “Opinion mining and sentiment analysis”, *Foundations and Trends in Information Retrieval*, vol. 2–1-2, 2008, pp. 1–135.

[52] Pang, B.; Lee, L.; Vaithyanathan, S. “Thumbs up?: sentiment classification using machine learning techniques”. In: ACL Conference on Empirical Methods in Natural Language Processing, 2002, pp. 79–86.

[53] Patcha, A.; Park, J.-M. “An overview of anomaly detection techniques: Existing solutions and latest technological trends”, *Computer networks*, vol. 51–12, 2007, pp. 3448–3470.

[54] Patist, J. P. “Optimal window change detection”. In: IEEE International Conference on Data Mining Workshops, 2007, pp. 557–562.

[55] Read, J. “Using emoticons to reduce dependency in machine learning techniques for sentiment classification”. In: ACL Student Research Workshop, 2005, pp. 43–48.

[56] Shannon, C. E. “A mathematical theory of communication”, *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5–1, 2001, pp. 3–55.

[57] Silva, I. S. “Análise adaptativa de fluxos de sentimento”. (Dissertação de Mestrado), Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais, UFMG, 2012, 88p.

[58] Silva, J. A.; Faria, E. R.; Barros, R. C.; Hruschka, E. R.; Carvalho, A. C. d.; Gama, J. “Data stream clustering: A survey”, *ACM Computing Surveys*, vol. 46–1, 2013, pp. 1–31.

[59] Street, W. N.; Kim, Y. “A streaming ensemble algorithm (sea) for large-scale classification”. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001, pp. 377–382.

[60] Thelwall, M.; Buckley, K.; Paltoglou, G. “Sentiment in twitter events”, *Journal of the American Society for Information Science and Technology*, vol. 62–2, 2011, pp. 406–418.

[61] Tsymbal, A.; Pechenizkiy, M.; Cunningham, P.; Puuronen, S. “Dynamic integration of classifiers for handling concept drift”, *Information fusion*, vol. 9–1, 2008, pp. 56–68.

[62] Tsytsarau, M.; Palpanas, T. “Survey on mining subjective data on the web”, *Data Mining and Knowledge Discovery*, vol. 24–3, 2012, pp. 478–514.

[63] Twitter. “About twitter”. Source: https://about.twitter.com/pt/company, February 02, 2016.

[64] Wang, H.; Can, D.; Kazemzadeh, A.; Bar, F.; Narayanan, S. “A system for real-time twitter sentiment analysis of 2012 us presidential election cycle”. In: ACL System Demonstrations, 2012, pp. 115–120.

[65] Wang, H.; Fan, W.; Yu, P. S.; Han, J. “Mining concept-drifting data streams using ensemble classifiers”. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003, pp. 226–235.

[66] Wenerstrom, B.; Giraud-Carrier, C. “Temporal data mining in dynamic feature spaces”. In: International Conference on Data Mining, 2006, pp. 1141–1145.

[67] Whitehead, M.; Yaeger, L. “Sentiment mining using ensemble classification models”. In: Innovations and Advances in Computer Sciences and Engineering, 2010, pp. 509–514.

[68] Yang, Y.; Pedersen, J. O. “A comparative study on feature selection in text categorization”. In: International Conference on Machine Learning, 1997, pp. 412–420.

[69] Zhou, Y.; Mulekar, M. S.; Nerellapalli, P. “Adaptive spam filtering using dynamic feature spaces”, *International Journal on Artificial Intelligence Tools*, vol. 16–04, 2007, pp. 627–646.

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br