Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Caio Luiggy Riyoichi Sawada Ueno

Análise sobre o Fator Temporal em Tarefas de Quantificação com Dados Textuais

São Carlos
2023

Caio Luiggy Riyoichi Sawada Ueno

Análise sobre o Fator Temporal em Tarefas de Quantificação com Dados Textuais

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Metodologias e Técnicas de Computação

Orientador: Diego Furtado Silva

São Carlos
2023

Aos amigos e colegas, que me incentivaram todos os dias e ofereceram apoio nos momentos críticos.

Agradecimentos

Agradecimentos ao Departamento de Computação da Universidade Federal de São Carlos – UFSCar. Agradecimentos especiais aos professores Dr. André Gustavo Maletzke e Dr. Diego Furtado Silva por todo o apoio e as discussões sobre os temas abordados neste trabalho.

"Quem alcança seu ideal, precisamente com isso o ultrapassa."
(Nietzsche)

Resumo

A quantificação, área relativamente nova em aprendizado de máquina, se preocupa em estimar a distribuição das classes em um determinado conjunto de instâncias. Usualmente, tarefas de quantificação são resolvidas por meio de classificação, onde um classificador prediz a classe das instâncias dentro do conjunto e uma contagem simples dos rótulos preditos é feita - método conhecido como *Classify and Count*. Entretanto, esse método não apresenta bons resultados à medida que a distribuição das classes no conjunto de teste se distância da distribuição das classes no conjunto de treinamento do classificador. Por isso, métodos e modelos específicos têm sido desenvolvidos para a resolução de problemas de quantificação.

Em cenários de dados volumosos é comum realizar análises da perspectiva temporal. Por exemplo, em domínios textuais como postagens em redes sociais, que possuem uma grande quantidade de dados não estruturados sendo gerados a todo instante, é um desafio grande extrair informações de forma condensada. Além disso, o domínio apresenta características próprias que aumentam a complexidade da forma como essas informações são extraídas. Uma das principais demandas nesse domínio é sumarizar o que ou como se fala sobre determinados tópicos na plataforma. Para tanto, é possível utilizar técnicas de quantificação para resumir uma quantidade massiva de textos.

Este trabalho apresenta uma análise sobre o problema de quantificação em conjuntos de dados textuais distribuídos ao longo do tempo. Mais precisamente, este trabalho propõe avaliar como a estrutura temporal dos dados afeta a performance de modelos treinados para solucionar o problema da quantificação. Três formas distintas foram adotadas para se entender o impacto da passagem do tempo: treinar somente uma vez o modelo de quantificação; atualizar o modelo periodicamente, diminuindo assim a defasagem temporal; e, por fim, uma abordagem de *forecasting* de séries temporais, utilizando modelos de regressão. Os resultados mostram que existem peculiaridades nos conjuntos de dados avaliados neste trabalho e que, em determinados cenários, modelos considerados estado-da-arte não apresentam as melhores performances.

Palavras-chave: Aprendizado de Máquina. Quantificação. Séries Temporais.

Abstract

The quantification task, a recently discovered field in machine learning, estimates the class distribution of a dataset. Usually, quantification tasks are solved through classification, an inducted classifier predicts each instance on the set and then counts how many were labeled for each class - this approach is also known as Classify and Count. However, the Classify and Count approach shows poor results as soon as the class distribution of the test set differs from the class distribution of the training set. Thus, specific algorithms and models have been proposed to solve quantification problems accurately.

It is really common to analyze big data through time. In text domains, as the *Twitter* platform, which have a large set of unstructured data being generate at every instant, it is challenging to extract useful and summarized information at the same time. Besides, text domains show specific characteristics that increase the complexity of how those information are extracted. A popular analysis is to discovery trending topics or how people's opinion about a specific topic. To do this, it is possible to use quantification methods to categorize and consequently summarize a massive number of texts.

The proposal of this work is to make an analysis about textual quantification problems distributed over time. More precisely, this work intent to evaluate how time affects the perfomance of quantification models. Three different approaches were evaluated to understand the impact of time: training only once the quantification model; update the model periodically, thus decreasing its time lag; and a forecasting approach, using regression models. This research presents some intereseting conclusions which show that there are some peculiarities in these evaluated datasets and that state-of-the-art models may not present the best performances as expected.

Keywords: Machine Learning. Quantification. Time Series.

Lista de ilustrações

Lista de tabelas

Sumário

Capítulo 1

Introdução

Em determinados problemas de *machine learning*, o objetivo final está em estimar corretamente a quantidade de instâncias pertencentes às classes conhecidas, e não somente o rótulo de cada uma das instâncias de forma individual (FORMAN, 2005), tarefa que recebe o nome de quantificação.

Por exemplo, Liu et al. (2021) apresentam um estudo de caso analisando a proporção de *posts* positivos diários de uma rede social antes e durante a pandemia de COVID-19 na China. Nessa análise, percebeu-se que o número de posts positivos caiu drasticamente após o início da pandemia e que o número de posts negativos (de tristeza ou medo) subiram. Consequentemente, concluiu-se que a pandemia afetou os indivíduos também de forma psicológica (LI et al., 2020).

Similarmente, Moraes, Manssour e Silveira (2015) fazem o mesmo tipo de análise de sentimento para *tweets* sobre o fatídico jogo de futebol em que a Alemanha eliminou a seleção brasileira na Copa do Mundo de 2014. A Figura 1 demonstra a polaridade dos textos ao longo do tempo. Em uma análise a nível de instâncias, não seria possível ter uma visão macro de como os sentimentos se alteraram durante o jogo e a popularidade de cada sentimento. Nessa visualização, inclusive, é possível perceber que há picos de *tweets* negativos no primeiro e quinto gols sofridos pelo Brasil, e que a partir do segundo gol da Alemanha, os *tweets* positivos deixam de existir. Ou seja, é possível traçar o comportamento e a evolução dos sentimentos ao longo do tempo.

Para que se tenha uma análise mais precisa, é necessário quantificar os dados da melhor forma possível - não necessariamente via classificação. Uma alternativa para esse problema é adotar uma estratégia própria de quantificação, induzindo um modelo que tenha como objetivo final predizer a distribuição das categorias existentes. A tarefa de quantificação foi formalmente introduzida por Forman (2005), e o seu objetivo principal

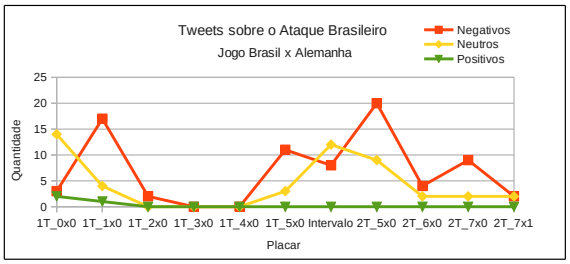


Figura 1 – Polaridade dos *tweets* ao longo de um jogo de futebol entre as seleções do Brasil e da Alemanha.

Fonte: (MORAES; MANSSOUR; SILVEIRA, 2015)

é estimar a proporção de instâncias pertencentes a cada uma das classes conhecidas em um conjunto de dados. Um exemplo típico de aplicação envolvendo quantificação é se estimar a proporção de textos com sentimentos positivos, neutros e negativos sobre um determinado assunto - produto, serviço, pessoa, etc - durante um determinado intervalo de tempo, principalmente em redes sociais (MORAES; MANSSOUR; SILVEIRA, 2015; LIU et al., 2021).

Muitas vezes, uma tarefa de quantificação é resolvida por meio da abordagem de classificação, uma vez que, obtido um classificador, é possível realizar a predição de cada uma das instâncias e a contar os rótulos atribuídos pelo modelo. Esse método é conhecido como *Classify and Count*. Entretanto, como demostrado em Forman (2005), o *Classify and Count* apresenta um viés sistêmico devido ao objetivo de classificadores de maximizar a acurácia. O conjunto de treinamento é parte fundamental desse viés, e conforme o conjunto predito por modelos de classificação se distancia do conjunto de treino em relação a distribuição das classes maior é o erro causado.

Além disso, trabalhos relacionados à quantificação usualmente focam numa configuração de *batch* único, ou seja, conjuntos de dados estáticos que não apresentam uma mudança de comportamento ou de conceito (FORMAN, 2005; SCHUMACHER; STROHMAIER; LEMMERICH, 2021; MOREO; SEBASTIANI, 2022). Nesse tipo de abordagem, o modelo é treinado com um conjunto de treinamento uma única vez e é utilizado para predizer outro conjunto independente de teste. Mesmo que muitos trabalhos apresentem uma configuração experimental que avalie mudanças pequenas e extremas na distribuição das classes, essa avaliação é feita de forma artificial e não necessariamente corresponde à natureza dos dados. Entretanto, problemas reais de quantificação podem apresentar conjuntos de teste com uma diferença na distribuição de classes significativa em relação ao treino. Usando o exemplo das redes sociais, a polaridade dos textos pode apresentar distribuições distintas em dias diferentes, ou seja, os próprios dados apresentam uma mudança de prevalência quando sua estrutura temporal é considerada.

Dessa forma, há poucos trabalhos na literatura que exploram essa característica presente em alguns conjuntos de dados. Essa escassez está muito associada à dificuldade em encontrar dados que possuam meta-informações relevantes, como a referência de tempo

de cada instância.

A proposta deste trabalho é estudar como modelos de quantificação, usualmente avaliados em uma configuração de *batch*, se comportam quando os dados estão dispostos em tempos diferentes, podendo apresentar mudanças na distribuição das classes ou até mesmo conceituais. Especificamente, o foco será em dados textuais de mídias sociais, como por exemplo o *Twitter* [1], que são textos curtos que geram um volume massivo de dados. Nesse cenário, este trabalho reúne experimentos que podem esclarecer como a tarefa de quantificação se comporta com dados distribuídos no tempo.

1.1 Objetivo e Hipótese

Este trabalho visa investigar mais profundamente alguns aspectos das tarefas de quantificação no domínio textual quando dispostas ao longo do tempo. E para tal, visa responder a seguinte questão de pesquisa:

- ❑ **QP**: Como o fator temporal impacta em tarefas de quantificação no domínio de dados textuais?

Para responder a **Questão de Pesquisa**, serão conduzidos alguns experimentos avaliando a performance de modelos de quantificação em conjuntos de dados que estão distribuídos ao longo do tempo e como essa performance se comporta. A hipótese é de que os quantificadores sofrem uma defasagem natural com o decorrer do tempo e consequentemente suas performances diminuem.

1.2 Organização

O texto está organizado da seguinte forma: no capítulo 2 é apresentada a fundamentação teórica por trás da tarefa de quantificação, bem como um apanhado de algoritmos disponíveis na literatura e formas de avaliação de quantificadores. No capítulo 3 estão dispostos trabalhos relacionados à quantificação. O capítulo 4 descreve todo o processo experimento experimental deste trabalho, desde os recursos utilizados até os algoritmos a serem avaliados. Em seguida, o capítulo 5 apresenta os resultados obtidos por meio das abordagens adotadas neste trabalho. Por fim, são apresentadas as conclusões deste trabalho juntamente com sugestões de trabalhos futuros.

[1] Recentemente renomeado para X.

Capítulo 2

Fundamentação Teórica

Neste capítulo serão descritos alguns conceitos sobre a tarefa de quantificação, na Seção 2.1, seguidos pela apresentação de algoritmos utilizados na literatura, na Seção 2.2 e, por fim, formas de avaliação na, Seção 2.3.

2.1 Quantificação

A primeira definição formal de quantificação foi apresentada por Forman (2005): Dado um conjunto de treinamento, a tarefa de quantificação se resume em induzir um quantificador para estimar a distribuição de classe de um outro conjunto de instâncias não rotuladas.

Considere um conjunto de instâncias rotuladas $T = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, onde $x_i \in \mathbb{R}^m$ (espaço de características) e $y_i \in C = \{c_1, c_2, ..., c_k\}$, onde C é o conjunto de classes conhecidas. Seja $A = \{x_1, x_2, ..., x_p\}$ um outro conjunto de instâncias não rotuladas disjunto de T. Um quantificador Q é um modelo que, treinado com as instâncias de T, é capaz de predizer a prevalência de cada classe em A de acordo com a Equação 1:

$$Q : A \rightarrow [0, 1]^k \tag{1}$$

onde k é o número de classes definidas. O objetivo é estimar uma distribuição de classes $[\hat{p}_1, \hat{p}_2, ..., \hat{p}_k]$, onde $\sum_{i=1}^{k} \hat{p}_i = 1$, que melhor se aproxima da real distribuição $[p_1, p_2, ..., p_k]$.

Essa definição implica em algumas diferenças em relação a outras tarefas de aprendizado supervisionado. As tarefas de classificação e regressão têm por objetivo modelar uma função que relaciona um vetor de variáveis independentes a uma variável dependente

- seja um rótulo ou um valor numérico. No exemplo acima, no momento da avaliação, tanto classificador quanto regressor fariam uma predição para cada vetor x de A individualmente. Ou seja, seriam necessárias p predições distintas, onde p é o número de instâncias em A. Adicionalmente, em ambos os casos, existe a premissa de que todas as instâncias são independentes e que os conjuntos de treino e teste são identicamente distribuídos (i.i.d) (CHAPELLE; CHI; ZIEN, 2006; HAN; KAMBER; PEI, 2012).

Entretanto, a quantificação não atua no mesmo nível de granularidade que as tarefas anteriores. O quantificador realiza uma única predição para todo o conjunto de teste. Além disso, em tarefas de quantificação, o objetivo principal é justamente estimar a distribuição de classes. Assumir que os conjuntos de treino e teste são identicamente distribuídos torna o problema trivial, pois a distribuição das classes seria constante em ambos os conjuntos. Portanto, é esperado que em problemas reais de quantificação haja uma diferença na prevalência entre os conjuntos.

2.2 Algoritmos de Quantificação

González et al. (2017) reúnem trabalhos pioneiros e categorizam as abordagens dos algoritmos de quantificação em três grupos: **Classify, Count, and Correct** descritos na Subseção 2.2.1; **Algoritmos de Classificação Tradicionais Adaptados** na Subseção 2.2.2; e, por fim, ***Matching* de Distribuições** na Subseção 2.2.3.

2.2.1 Classify, Count, and Correct

Nesta categoria de algoritmos, estão agrupados os métodos que se baseiam em classificar instâncias, realizar a contagem dos rótulos preditos e aplicar, ou não, uma correção.

2.2.1.1 *Classify and Count*

O *Classify and Count* (CC) (FORMAN, 2005) é um método simples e direto para quantificar um conjunto de instâncias. Com um classificador base, o método realiza uma predição para cada instância e conta quantas instâncias foram rotuladas em cada classe conhecida. É fácil notar que em um cenário onde o classificador é ótimo, a contagem, ou quantificação, dos rótulos também é ótima. Entretanto, dificilmente é possível obter um modelo ótimo em problemas reais.

Uma característica vantajosa em um problema binário de quantificação, é que os falsos positivos podem compensar, em partes ou na totalidade, os falsos negativos. Isso pode ser visto na Equação 2, que ilustra a predição do CC, denotada por $\hat{p}$, com base na porcentagem real de positivos, p, e nas taxas de tpr e fpr do classificador base:

$$\hat{p}(p) = p \cdot tpr + (1 - p) \cdot fpr \tag{2}$$

Porém, o principal problema dessa abordagem é que o viés do classificador é propagado para a quantificação, pois a contagem depende justamente das taxas de *tpr* (verdadeiro positivo) e *fpr* (falso positivo) do classificador base utilizado, como indicado pela Equação 2. No caso binário, o classificador vai subestimar a contagem da classe positiva caso ele tenda a errar mais exemplos da classe positiva. Esse erro na contagem tende a aumentar conforme a porcentagem de exemplos positivos no conjunto de teste se distancia da porcentagem positiva no conjunto de treinamento (FORMAN, 2008).

Além das taxas de *tpr* e *fpr* em si, há também o *threshold* aplicado ao *score* do classificador, que distingue uma predição como sendo pertencente a uma classe ou outra. Usualmente, esse limiar é definido como $0,5$. Note que o *threshold* impacta diretamente as taxas de *tpr* e *fpr*, pois é com base nele que se atribui um determinado rótulo ou não para cada instância predita.

2.2.1.2 *Adjusted Classify and Count*

O *Adjusted Classify and Count* (FORMAN, 2005), ou ACC, é uma extensão do CC, em que, após a etapa de classificação, é feito um ajuste na contagem baseado em valores estimados de *tpr* e *fpr* do classificador base. Geralmente, essas taxas são estimadas por meio de um conjunto de validação. O sistema de Equações 3 apresenta as informações necessárias para ajustar a contagem de um classificador base em um problema binário:

$$\begin{cases} TP + FP = \hat{p} \\ TP = tpr \cdot p \\ FP = fpr \cdot (1 - p) \end{cases} \tag{3}$$

onde TP é a quantidade de instâncias positivas rotuladas de forma correta (verdadeiros positivos); FP é a quantidade de instâncias negativas rotuladas erroneamente (falsos positivos); p é a porcentagem real de exemplos positivos; e $\hat{p}$ a porcentagem de instâncias rotuladas positivamente pelo classificador. Note que para FP, em vez da porcentagem de instâncias negativas, multiplica-se pelo complemento da porcentagem real dos positivos. Por substituição e resolvendo para p obtém-se a Equação 4:

$$p = \frac{\hat{p} - fpr}{tpr - fpr} \tag{4}$$

Embora, teoricamente, a Equação 4 retorne a prevalência exata para a classe positiva, as taxas de *tpr* e *fpr* são suscetíveis a ruído e dificilmente são perfeitas. Além disso, a equação pode retornar valores irreais - fora do intervalo [0, 1] - o que geralmente é corrigido retornando o valor limite mais próximo. Outra característica do ACC é a necessidade de separar um conjunto adicional de validação para realizar a estimativa das taxas de *tpr* e *fpr*, ou até mesmo métodos mais complexos de validação cruzada.

2.2.1.3 *Probabilistic Classify and Count*

As abordagens anteriores (CC e ACC) possuem como hiperparâmetro o *threshold* que demarca o que é considerado uma instância positiva ou não, geralmente baseado na probabilidade que o classificador base retornou. A depender da escolha, a quantidade de exemplos considerados positivos e negativos pode mudar consideravelmente e, consequentemente, o cálculo das taxas *tpr* e *fpr* também, como discutido anteriormente.

O método *Probabilistic Classify and Count* (BELLA et al., 2010) remove a necessidade da escolha de um *threshold* e se baseia somente na probabilidade dada pelo classificador. Dado um conjunto de teste, o classificador estima a probabilidade de cada exemplo pertencer à classe positiva - no caso binário. Para estimar a prevalência, é calculada a média das probabilidades para cada classe, como mostra a Equação 5.

$$\hat{p}(\oplus) = \frac{1}{n} \cdot \sum_{n=1}^{n} \hat{P}(\oplus|x_n) \tag{5}$$

$\hat{P}(\oplus|x_n)$ é a probabilidade estimada pelo classificador base da instância x_n ser positiva e n é a quantidade de instâncias no conjunto predito. No trabalho de Bella et al. (2010) é chamado de *Probability Average* (PA), mas também é encontrado como *Probabilistic Classify and Count* (PCC) na literatura (GONZáLEZ et al., 2017; ESULI; SEBASTIANI, 2015; GAO; SEBASTIANI, 2016; SCHUMACHER; STROHMAIER; LEMMERICH, 2021; MALETZKE et al., 2021). Similiar ao CC, é necessário apenas ter induzido um classificador base que retorne as probabilidades condicionais de cada instância para estimar a prevalência de um conjunto por meio do PCC.

Adicionalmente, o PCC requer um classificador base calibrado (BELLA et al., 2010), o que pode adicionar uma etapa a mais no momento do treinamento do modelo a depender da implementação.

2.2.1.4 *Probabilistic Adjusted Classify and Count*

O PCC também pode ser ajustado por meio das taxas médias de probabilidades estimadas, proposto como *Scaled Probability Average* (BELLA et al., 2010), ou *Probabilistic Adjusted Classify and Count* (PACC). Considerando $\pi(\oplus|\oplus)$ como a média das probabilidades estimadas para exemplos positivos e $\pi(\oplus|\ominus)$ a média das probabilidades estimadas para exemplos negativos, utilizando o mesmo racional do ACC (BELLA et al., 2010), obtem-se a seguinte Equação 6:

$$p = \frac{\hat{p} - \pi(\oplus|\ominus)}{\pi(\oplus|\oplus) - \pi(\oplus|\ominus)} \tag{6}$$

O PACC, ou *Scaled Probability Average*, também pode retornar valores fora do intervalo [0, 1] e necessita de uma correção, ajustando valores negativos para 0 e valores acima de 1 para 1. Além disso, faz parte do método estimar a média de probabilidades $\pi(\oplus|\oplus)$

e $\pi(\oplus|\ominus)$ por meio de um ou mais conjuntos de validação. Também pode envolver uma etapa de calibração do classificador base como apontado em Bella et al. (2010).

2.2.1.5 Seleção de *Threshold*

Quanto maior a discrepância entre a prevalência do treino e do teste, pior tende ser as estimativas do quantificador ACC (FORMAN, 2005). A Figura 2 ilustra esse comportamento, onde no eixo x temos a porcentagem de instâncias positivas no teste e no eixo y o erro absoluto. A quantidade de exemplos positivos no treino é fixa, porém cada gráfico apresenta essa relação com uma quantidade de instâncias negativas no treino diferente, N, aumentando da esquerda para a direita. Ou seja, quanto mais balanceado for o conjunto de treino do ACC, menos suscetível ele será em relação a mudança de prevalência do teste.

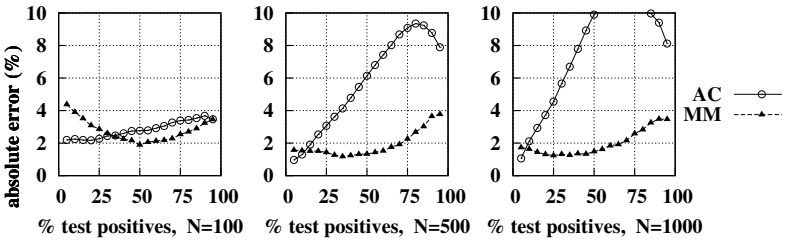


Figura 2 – O ACC tende a piorar as estimativas em conjuntos de teste onde a prevalência se distancia da prevalência do treino.

Fonte: (FORMAN, 2005)

Novamente, esse comportamento está relacionado as taxas *tpr* e *fpr* do classificador base a ser corrigido pelo ACC, que são impactadas pela escolha do *threshold*. Num cenário extremo, onde a classe positiva é extremamente rara, a taxa de *tpr* pode ser igual a 0.

Outros métodos propostos focam em encontrar um valor de *threshold* que otimize o *trade-off* entre as taxas de *tpr* e *fpr* (FORMAN, 2006).

1. **X**: *threshold* que satisfaça $fpr = 1 - tpr$;
2. **MAX**: *threshold* que maximize a diferença entre *tpr* e *fpr*, ou seja, $argmax(tpr - fpr)$;
3. **T50**: *threshold* que satisfaça $tpr = 50\%$;
4. **Median Sweep (MS)**: estima a prevalência para cada *threshold* utilizando a mesma abordagem do ACC - Equação 4 - e retorna a mediana;

Note que esses métodos também necessitam de um conjunto de validação para estimar as taxas de *tpr* e *fpr* por meio da busca de um valor de threshold que satisfaça a regra respectiva do método.

2.2.2 Algoritmos de Classificação Tradicionais Adaptados

Diferentemente dos modelos anteriores, os métodos descritos a seguir são específicos para a tarefa de quantificação. Ou seja, são algoritmos construídos para solucionar diretamente problemas de quantificação.

2.2.2.1 *Quantification Trees*

Milli et al. (2013) propõem a *Quantification Tree* (QT), um algoritmo que se utiliza da construção e estrutura de uma árvore de decisão para solucionar o problema de quantificação. O algoritmo constrói uma árvore dividindo os nós baseado em métricas de quantificação, ao invés de métricas tradicionais de classificação. Duas medidas são consideradas:

1. ***Classification Error Balancing***: Para cada *split* e para cada classe conhecida c_j, a métrica é calculada da seguinte forma: $E^1_{cj} = |FP_{cj} - FN_{cj}|$, onde FP_{cj} e FN_{cj} representam o número de falsos positivos e falsos negativos para a classe j, respectivamente. A quantificação é ótima quando $E^1_{cj} = 0$ - ou FP e FN são iguais a 0 ou FP = FN.

2. ***Classification-Quantification Balancing***: essa medida leva em consideração também a classificação em sua equação: $E^2_{cj} = |FP_{cj} - FN_{cj}| \cdot |FP_{cj} + FN_{cj}|$. Se $FP_{cj} + FN_{cj} = 0$, ou seja, a classificação é perfeita, então $E^2_{cj} = 0$.

Para cada *split*, o método armazena os valores em um vetor $E = [E_{c1}, E_{c2}, ..., E_{cj}]$, sendo E uma das métricas descritas anteriormente, E^1 ou E^2. O *score* do *split* é calculado por meio da métrica *L2-norm* de E, $||E||_2$. Por fim, é calculada a qualidade do *split* utilizando-se a equação de ganho de informação: $\Delta = ||E_{pai}||_2 - ||E_{filho}||_2$. Quanto maior o Δ, melhor é o *split*. A condição de parada é $\Delta \leq 0$.

Os autores também proprõem a construção de uma *Random Forest*, um conjunto de *Quantification Trees* para formar um *ensemble* de quantificadores base. A predição é feita por meio da média das predições individuais de cada *Quantification Tree*.

2.2.2.2 SVM(KLD)

Esuli e Sebastiani (2015) proprõem um método que, dada uma medida de performance, treina um modelo de classificação otimizando essa medida. A proposta dos autores é utilizar a métrica *Kullback-Leibler Divergence* (KLD) com o algoritmo SVM_{perf} (JOACHIMS, 2005). O SVM_{perf} pertence a família de *Support Vector Machines* que podem gerar classificadores otimizados para uma função não-linear e multi-variada, como é o caso da métrica KLD. Isso é interessante pois, na quantificação, o erro de uma instância individual pode impactar positiva ou negativamente o erro geral dependendo de como outra instância é classificada.

Os autores argumentam que usar uma função como a KLD, que é focada inteiramente para a tarefa de quantificação, remove o viés de otimizar a acurácia do classificador base, no caso a SVM. Dessa forma, seria possível obter um bom quantificador, mesmo que a acurácia fosse prejudicada. Um exemplo é se quando os falsos positivos são iguais aos falsos negativos ($FP = FN$), onde a quantificação possui um valor ótimo mesmo que o modelo classifique erroneamente instâncias.

2.2.3 *Matching* de Distribuições

Neste grupo estão dispostos modelos que tentam aproximar a distribuição de *scores* do conjunto de treino com a distribuição do conjunto de teste, minimizando uma função de distância. Esses *scores* são produzidos por meio de um classificador base induzido.

2.2.3.1 *Mixture Model* de Forman (2005)

Forman (2005) é o primeiro a propôr um algoritmo que se baseia em modelar a distribuição dos *scores* calculados no conjunto de teste (D_t) como uma composição de distribuições de *scores* dos exemplos positivos (D_+) e negativos (D_-) estimados por meio de um conjunto de validação. A Equação 7 demonstra essa composição:

$$total \cdot D_t = p \cdot D_+ + n \cdot D_- \tag{7}$$

onde *total* representa a quantidade total de exemplos no teste e p e n a quantidade de exemplos positivos e negativos, respectivamente. Para as distribuições, Forman (2005) optou por usar a função de distribuição acumulada, ou *cumulative distribution function* (CDF). Para estimar a prevalência do conjunto de treino, é realizada uma busca pela composição das distribuições que melhor se assemelha à D_t, variando o valor de p de 0% a 100% (intervalo de valores de prevalência possíveis), em passos de 0,5%. Como $n = (1-p)$ no caso binário, variar p também mexe no valor de n.

2.2.3.2 HDy

González-Castro, Alaiz e Alegre (2013) apresentam um algoritmo que modela as distribuições de *scores* por meio de um histograma. Uma vez obtidas as distribuições de *scores* das classes positiva (V^+) e negativa (V^-), via um classificador base e um conjunto de validação, o HDy calcula o fator α que minimiza a distância entre a distribuição composta V e a distribuição T gerada a partir do conjunto teste - mesmo racional apresentado em 2.2.3.1. A distância utilizada pelo HDy é a distância *Hellinger* (CSISZÁR; SHIELDS et al., 2004), onde cada histograma, gerado a partir de sua distribuição respectiva, é representado por um vetor em $\mathbb{R}^b$ - sendo b o número de *bins*.

O Sistema de Equações 8 apresenta a composição pelo fator α do conjunto de validação e o método usado pelo HDy para estimar a prevalência da classe positiva, respectivamente. HD representa o cálculo da distância *Hellinger*.

$$\begin{cases} V = \alpha \cdot V^{+} + (1 - \alpha) \cdot V^{-} \\ HDy(V, T) = \underset{0 \leq \alpha \leq 1}{\arg\min}\{HD(V, T)\} \end{cases} \tag{8}$$

González-Castro, Alaiz e Alegre (2013) estimam o fator α por meio de uma busca linear no intervalo [0, 1]. Note que α é a própria estimativa da prevalência da classe positiva e $(1 - \alpha)$ a estimativa para a classe negativa. Além disso, González-Castro, Alaiz e Alegre (2013) estimam a prevalência de um único conjunto de teste usando 11 quantidades de *bins* diferentes e retornam a mediana das estimativas como predição final.

2.2.3.3 *Distribution y-Similarity*

Em Maletzke et al. (2019), o *Distribution y-Similarity* (DyS) é proposto como um *framework* que generaliza o método do HDy. O DyS permite outras funções de distância ou dissimilaridade entre distribuições. A Equação 9 a seguir ilustra como o *framework* DyS estima a prevalência da classe positiva:

$$DyS(S^{+}, S^{-}, Z) = \underset{0 \leq \alpha \leq 1}{\arg\min}\{D(\alpha \cdot H(S^{+}) + (1 - \alpha) \cdot H(S^{-}), H(Z))\} \tag{9}$$

onde S^{+}, S^{-} e Z são as distribuições dos *scores* dos exemplos positivos, negativos e do conjunto de teste, respectivamente, D é uma função de dissimilaridade e H é uma função que transforma uma amostra de *scores* em uma representação compatível com a função de dissimilaridade escolhida. A generalização se encontra nas funções D e H, que precisam apenas satisfazer as propriedades de calcular a dissimilaridade entre histogramas e transformar amostras em histogramas.

Os autores reportam que a função de dissimilaridade Topsøe (CHA, 2008) apresentou os menores erros nos experimentos conduzidos. Ao longo deste trabalho, essa abordagem em conjunto com a função de dissimilaridade Topsøe será referenciada como DyS-Topsøe, enquanto que o termo DyS irá representar o *framework*. Assim como o método HDy, Maletzke et al. (2019) também estimam a prevalência de um conjunto de teste variando a quantidade *bins* dos histogramas e retornam a mediana dessas múltiplas predições.

2.2.3.4 SORD

Maletzke et al. (2019) propõem um algoritmo baseado em uma função de dissimilaridade que não depende da represetação das distribuições por meio de histogramas, baseada num caso especial da distância ORD (CHA; SRIHARI, 2002). É uma variação do DyS que atua diretamente com os *scores* de cada distribuição e calcula o custo necessário da

transformação de uma na outra sem a necessidade de uma função de transformação H. A Equação 10 ilustra a predição do método:

$$\hat{p} = \underset{0 \leq \alpha \leq 1}{\arg\min}\{SORD(S^+, S^-, \alpha, Z)\} \tag{10}$$

Como essa abordagem não depende de histogramas, não há o parâmetro da quantidade de *bins*, como nos métodos HDy e DyS-Topsøe, e, consequentemente, não há uma suavização da predição por meio de múltiplas predições por meio de tamanhos de *bins* diferentes.

2.2.3.5 *Sample Mean Matching*

Hassan, Maletzke e Batista (2020) propõem o *Sample Mean Matching* (SMM), um método mais simples para o *framework* DyS, onde as distribuições dos *scores* são representadas pelas médias respectivas, e não por histogramas. A Equação 11 demonstra o cálculo da prevalência pelo SMM:

$$\hat{p} = \underset{0 \leq \alpha \leq 1}{\arg\min}\{|\alpha \cdot m^+ + (1 - \alpha) \cdot m^- - m^t|\} \tag{11}$$

onde m^+, m^- e m^t representam a média dos *scores* dos exemplos positivos, negativos e do conjunto de teste, respectivamente. O racional por trás do SMM é que a média de uma composição de duas amostras é uma soma ponderada das médias individuais de cada amostra - ou seja, uma média ponderada. A Equação 12 demonstra essa modelagem:

$$\alpha \cdot m^+ + (1 - \alpha) \cdot m^- = m^t \tag{12}$$

Entretanto, é possível simplificar a Equação 12 resolvendo para α utilizando a seguinte Equação 13:

$$\alpha = \frac{m^t - m^-}{m^+ - m^-} \tag{13}$$

Diferentemente das outras abordagens citadas anteriormente, HDy e DyS, o SMM não necessita de uma busca para encontrar um valor satisfatório para o α. Além disso, não depende de uma representação vetorial, como um histograma, das distribuições de *scores*, e consequentemente não é influenciado pela dimensionalidade dessas representações. Dessa forma, o SMM é o quantificador baseado em *Mixture Model* mais rápido quando comparado a outros métodos dentro da categoria de *matching* de distribuições (HASSAN; MALETZKE; BATISTA, 2020).

2.3 Avaliação de Quantificadores

Hassan, Maletzke e Batista (2021) discutem sobre as características dos dois protocolos utilizados para avaliar quantificadores: o *Artificial-Prevalence Protocol* (APP) e o *Natural-Prevalence Protocol* (NPP). A Figura 3 ilustra o funcionamento de cada um deles.

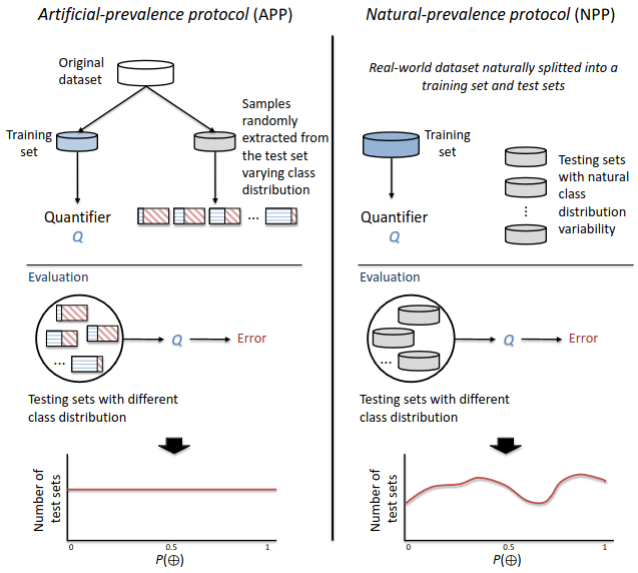


Figura 3 – Comparação dos dois protocolos mais utilizados para avaliação de modelos de quantificação, APP à esquerda e NPP à direita.

Fonte: (HASSAN; MALETZKE; BATISTA, 2021)

O APP, inicialmente proposto por Forman (2005), é o método mais utilizado para avaliar algoritmos de quantificação, com algumas variações dependendo do trabalho (FORMAN, 2006; MOREO; SEBASTIANI, 2022; BELLA et al., 2010; MALETZKE et al., 2019; HASSAN; MALETZKE; BATISTA, 2020; SCHUMACHER; STROHMAIER; LEMMERICH, 2021). O principal objetivo é criar múltiplos conjuntos de teste que possuem diferentes prevalências, porém de maneira uniforme. Note que a característica de ser uniforme nem sempre é satisfeita, como em Schumacher, Strohmaier e Lemmerich (2021). Para um problema de quantificação binário, o APP cria conjuntos de teste seguindo o espectro possível para a distribuição da classe positiva, por exemplo: $\{0, .01, .02, ..., .99, 1\}$. A quantidade de valores utilizados ou o passo entre eles variam de acordo com o trabalho. Geralmente esse processo de geração de subconjuntos de teste é feito via amostragem de um conjunto suficientemente grande, utilizando a técnica de *undersampling* (HASSAN; MALETZKE; BATISTA, 2021).

Dado essa configuração, Hassan, Maletzke e Batista (2021) discorrem sobre duas limitações do protocolo APP: (1) o espectro de valores possíveis para a distribuição da classe positiva é tratada de forma discreta; (2) esse espectro de valores também segue uma distribuição uniforme.

Sobre a abordagem discreta do espectro de valores, modelos como HDy e DyS-Topsøe, por fazerem uma busca no intervalo $[0, 1]$ de prevalências possíveis, podem ter uma certa vantagem durante o cálculo das dissimilaridades. Os autores demonstram que se a cardinalidade desse intervalo de busca for relativamente maior do que a cardinalidade de prevalências distintas nos conjuntos de teste do APP, é possível obter performances ilusórias a cerca desses modelos. Adicionalmente, recomendam que, nos conjuntos de teste, haja pelo menos 100 distribuições distintas.

Em relação à característica uniforme das distribuições utilizadas no APP, Hassan, Maletzke e Batista (2021) apontam para possíveis conclusões errôneas ao se adotar o CC como um quantificador *baseline*. Ter uma distribuição uniforme implica que, na média, p é 0.5 (50%). Formalmente, o valor esperado de p nos conjuntos de testes gerados pelo protocolo é $\mathbb{E}(p) = 0.5$ - num problema de quantificação binário. Quantificadores que tendem suas predições para o valor esperado são menos penalizados na média. Ou seja, eles se beneficiam da configuração uniforme do protocolo. Para aumentar a percepção sobre esse problema, propõem um quantificador nomeado *Lazy* que sempre retorna $\mathbb{E}(p)$ para ser utilizado como *baseline*. Dessa forma, acreditam que qualquer quantificador, para ser considerado útil, deve obter uma performance melhor do que o *Lazy*.

O NPP consiste em obter conjuntos de dados com múltiplos subcojuntos individuais coletados de forma natural. Entretanto, conjuntos de dados com essa característica são difíceis de se obter. Gao e Sebastiani (2016) fazem uma avaliação de quantificadores utilizando um conjunto de dados NPP no domínio textual, entretanto ressaltam que a quantidade dos conjuntos de teste é muito baixa e que a variabilidade das distribuições de classe é pequena e optam por reavaliar os modelos por meio do APP (MOREO; SEBASTIANI, 2022).

Além da variação da distribuição de classes do teste e do treino, Schumacher, Strohmaier e Lemmerich (2021) analisam o impacto do tamanho do treino - quantidade de informação disponível para o aprendizado dos modelos. Os autores concluem que, tanto para *datasets* binários quanto para o cenário multi-classe, a performance dos modelos tende a deteriorar conforme a quantidade de instâncias no treino diminui.

Maletzke et al. (2021) avaliam o impacto do tamanho do conjunto de teste no desempenho dos modelos de quantificação. Os autores discutem que muitos trabalhos na literatura não consideram a influência do tamanho do teste, de forma que esse impacto acaba diluído nos resultados. Algoritmos como o MS e o CC, que não apresentam um bom ranqueamento nos resultados gerais, estão no top 3 quando avaliados somente em conjuntos de testes pequenos - de 10 instâncias. Por fim, os autores concluem que variar o tamanho dos conjuntos de teste traz uma maior completude para a avaliação de modelos de quantificação.

2.4 Métricas

Tarefas de quantificação possuem métricas de performance assim como as tarefas de classificação e regressão. Nesta seção são descritas algumas métricas utilizadas para medir a performance de quantificadores.

2.4.1 Erro Absoluto Médio

O Erro Aboluto Médio (EAM) é a média das diferenças entre a prevalência predita e a prevalência real de cada classe - popularmente conhecido como *Mean Absolute Error* (MAE). É definido por meio da Equação 14 a seguir, onde k é o número de classes conhecidas:

$$MAE(p, \hat{p}) = \frac{1}{k} \sum_{i=1}^{k} |p_i - \hat{p}_i| \tag{14}$$

A métrica recai no intervalo fechado $[0, 1]$, o que facilita a comparação de modelos, inclusive entre *datasets* diferentes. Além disso, como descrito por Sebastiani (2020), é robusta à *outliers* e possui propriedades satisfatórias. Dado essas características, o EAM é frequentemente utilizado em trabalhos sobre quantificação (CSISZÁR; SHIELDS et al., 2004; GAO; SEBASTIANI, 2016; MALETZKE et al., 2019; MALETZKE et al., 2021).

2.4.2 Kullback-Leibler Divergence Normalizado

Kullback-Leibler Divergence (KLD) é outra métrica muito usada em tarefas de quantificação (FORMAN, 2005; FORMAN, 2006; ESULI; SEBASTIANI, 2015). É definida pela Equação 15 a seguir:

$$KLD(p, \hat{p}) = \sum_{i=1}^{k} p_i \cdot \log \frac{p_i}{\hat{p}_i} \tag{15}$$

Entretanto, como aponta González et al. (2017), KLD não é realmente uma métrica, pois não satisfaz a desigualdade triangular e não é simétrica. Além disso, quando p_i ou $\hat{p}_i$ é igual a 0, o resultado é indefinido. Para contornar essa desvantagem, Gao e Sebastiani (2016) normalizam a métrica por meio de uma função logística definida pela Equação 16 - uma versão normalizada denominada KLD Normalizada, ou NKLD (SCHUMACHER; STROHMAIER; LEMMERICH, 2021).

$$NKLD(p, \hat{p}) = 2 \cdot \frac{e^{KLD(p,\hat{p})}}{1 + e^{KLD(p,\hat{p})}} - 1 \tag{16}$$

Dessa forma, a métrica permanece no intervalo $[0, 1]$, o que a torna mais interpretável - 0 sendo o valor ótimo e 1 divergência total. Além disso, esse ajuste torna mais fácil comparar o valor da métrica entre conjuntos de dados distintos.

2.4.3 Correlação de Pearson

Além de medir o erro da prevalência predita em relação a prevalência real para cada *batch* de teste, e devido a característica temporal dos dados, se faz necessário entender se o comportamento ao longo do tempo das predições de cada classe se assemelham ou não com o comportamento real. Nesse sentido, a correlação de *Pearson* é uma métrica que permite medir a similaridade do formato das distribuições ao longo do tempo. A Equação 17 a seguir demonstra o cálculo da métrica:

$$\rho = \frac{cov(X, Y)}{\sqrt{var(X) \cdot var(Y)}} \tag{17}$$

Através da correlação é possível entender duas características interessantes de séries temporais: tendência e sazonalidade. Ou seja, caso a prevalência de uma determinada classe esteja aumentando e a predição do quantificador também, mesmo que numericamente os valores sejam diferentes, isso significa que o modelo foi capaz de aprender a direção da mudança das prevalências. Essa característica pode ser interessante em cenários onde somente a tendência da distribuição, de forma geral, importa.

2.5 Testes de Hipótese

Embora tenha introduzido a tarefa de quantificação, Forman (2005) não realiza testes estatísticos para se certificar das diferenças obtidas entre os modelos avaliados. Trabalhos posteriores (MOREO; SEBASTIANI, 2022; ESULI; SEBASTIANI, 2015) utilizam um Teste t de *Student* para verificar as diferenças entre as ditribuições de erros dos quantificadores. Demsar (2006) discorre sobre três limitações do Teste t: (1) comensurabilidade das diferenças entre *datasets* diferentes; (2) as distribuições comparadas precisam possuir uma distribuição normal; (3) o teste é muito suscetível à *outliers*, que aumentam o desvio padrão e consequentemente dimimuem o poder do teste.

Em Hassan, Maletzke e Batista (2021), os autores utilizam o teste de *Friedman* ao comparar dois modelos de quantificação. Em outros trabalhos com múltiplos modelos a serem comparados (MALETZKE; REIS; BATISTA, 2017; MALETZKE et al., 2021; BELLA et al., 2010; SCHUMACHER; STROHMAIER; LEMMERICH, 2021; HASSAN; MALETZKE; BATISTA, 2020) é utilizada uma combinação de testes: primeiro é aplicado o teste de *Friedman* em todo o conjunto de modelos e depois o teste de *Nemenyi* em cada par de modelos existentes. O teste de *Friedman* compara a média do ranqueamento dos modelos nos conjuntos de teste, onde a hipótese nula é de que não há diferença entre seus ranqueamentos. Caso a hipótese nula seja rejeitada, é então utilizado um teste *post-hoc* como o de *Nemenyi*.

Outros trabalhos (GAO; SEBASTIANI, 2016; GONZáLEZ-CASTRO; ALAIZ; ALEGRE, 2013) optaram por aplicar o teste de *Wilcoxon* que utiliza a medida de performance

do par de modelos sendo comparados e, consequentemente, considera a magnitude da diferença observada, diferentemente do teste de *Nemenyi*. O conjunto de Equações 18 a seguir ilustra os cálculos do teste de *Wilcoxon*:

$$\begin{cases} T^+ = \sum\limits_{d_i>0} rank(d_i) + \frac{1}{2} \cdot \sum\limits_{d_i=0} rank(d_i) \\ T^- = \sum\limits_{d_i<0} rank(d_i) + \frac{1}{2} \cdot \sum\limits_{d_i=0} rank(d_i) \\ T = min(T^+, T^-) \\ z = \frac{T - 0.25 \cdot N \cdot (N+1)}{\sqrt{\frac{N \cdot (N+1) \cdot (2N+1)}{24}}} \end{cases} \tag{18}$$

Ismail-Fawaz et al. (2023) apresentam uma discussão em torno dos testes *Nemenyi* e *Wilcoxon*. Demsar (2006) propõe que o teste de *Wilcoxon* seja usado quando há somente dois modelos a serem comparados. Entretanto, o *Nemenyi* apresenta algumas limitações: (1) diferenças grandes e pequenas de performance são ignoradas uma vez que somente o ranqueamento é utilizado; (2) adicionar ou remover modelos impacta globalmente o teste, pois um único modelo é capaz de alterar o ranqueamento geral.

Para contornar esses problemas, Benavoli, Corani e Mangili (2016) argumentam para que o teste de *Wilcoxon* substitua o teste de *Nemenyi*. Além disso, os autores também discorrem que é necessário aplicar uma correção no nível de confiança quando múltiplos testes são realizados, de forma a controlar o erro do tipo I de comparações múltiplas.

Capítulo 3

Revisão da Literatura

Este capítulo apresenta os trabalhos relacionados mais relevantes ao tópico desta pesquisa. A Seção 3.1 aborda trabalhos sobre a tarefa de quantificação de forma geral, ressaltando os modelos considerados estado-da-arte; a Seção 3.2 descreve trabalhos sobre quantificação no domínio textual; e, por fim, a Seção 3.3 descreve trabalhos sobre quantificação temporal.

3.1 Quantificação

Esta seção apresenta dois trabalhos relacionados à tarefa de quantificação: o primeiro, na Subseção 3.1.1, é uma comparação da complexidade de tempo requerida por modelos considerados estado-da-arte juntamente com suas respectivas performances; e o segundo, na Subseção 3.1.2, apresenta uma avaliação de modelos no contexto de tarefas de quantificação binária e multi-classe.

3.1.1 *Accurately quantifying a billion instances per second*

Hassan, Maletzke e Batista (2020) apresentam um estudo em termos de complexidade de tempo de diversos modelos de quantificação considerados estado-da-arte. A Tabela 1 a seguir resume a complexidade de tempo de treinamento e predição de cada algortimo estudado no trabalho. Vale ressaltar que o modelo SMM - marcado na tabela com um asterisco - foi o algortimo proposto pelos autores.

Para a avaliação da performance dos quantificadores, os autores utilizaram o protocolo APP, discutido na Seção 2.3, com o acréscimo de variar também o tamanho dos conjuntos de testes gerados. Ou seja, para cada tamanho possível, foram gerados conjuntos de teste

Tabela 1 – Erro Absoluto Médio de cada quantificador.

modelo	treinamento	predição
CC	$O(M(T_r))$	$O(C(\|T_e\|))$
ACC	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|))$
PCC	$O(M(T_r) + \|T_r\|log\|T_r\|)$	$O(C(\|T_e\|) + log\|T_e\|)$
PACC	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})) + \|T_r\|log\|T_r\|))$	$O(C(\|T_e\|) + log\|T_e\|)$
X	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|))$
T50	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|))$
MAX	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|))$
MS	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|) + \|r_{t_h}\|)$
MS2	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|) + \|r_{t_h}\|)$
QT	$O(M(T_r))$	$O(C(\|T_e\|))$
EMQ	$O(M(T_r))$	$O(C(\|T_e\|) + \|T_e\|tl)$
DyS-Topsøe	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|) + b \cdot log(\frac{1}{E}))$
SORD	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|) + (\|T_r\| + \|T_e\|)log(\|T_r\| + \|T_e\|))$
HDy	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|) + b \cdot \|\alpha\|)$
SMM*	$O(O(M(T_r)) + k(M(\|T_r\| - \frac{\|T_r\|}{k}) + C(\frac{\|T_r\|}{k})))$	$O(C(\|T_e\|))$

Fonte: Adaptado de Hassan, Maletzke e Batista (2020)

variando a distribuição das classes, respeitando a uniformidade da distribuição de p na avaliação. O tamanho dos conjuntos de teste variou de 10 a 100, com um incremento de 10, e de 100 a 500, com um incremento de 100. A Figura 4 ilustra a comparação de tempo e performance dos quantificadores.

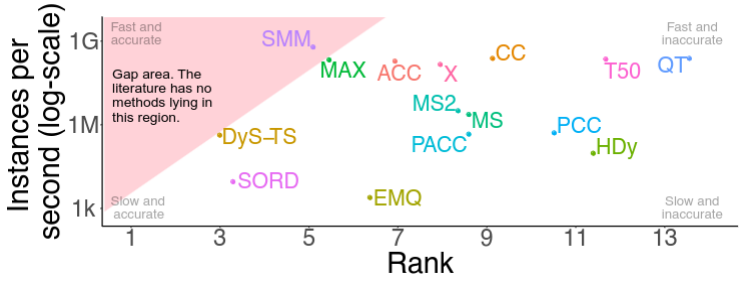


Figura 4 – Comparação do tempo e da performance dos 15 modelos avaliados em 25 *datasets* distintos.

Fonte: Hassan, Maletzke e Batista (2020)

Nota-se que muitos quantificadores possuem um parâmetro k em suas complexidades de tempo de treinamento. Esse parâmetro indica a quantidade de validações cruzadas realizadas para estimar artefatos internos necessários para os quantificadores, como taxas de *tpr* e *fpr* para o ACC ou distribuições de *scores* de exemplos positivos e negativos para o HDy ou DyS. Consequentemente, muitos dos algoritmos, especialmente os que utilizam um classificador base em sua composição, apresentam uma complexidade de tempo de treinamento similar devido à essa necessidade.

A diferença entre os algoritmos se faz mais presente no tempo de predição de um conjunto de teste. Modelos mais simples, como o ACC, X, T50 e MAX, necessitam apenas da predição do classificador base, $O(C(|T_e|))$, e um ajuste no resultado do classificador,

que possui tempo constante em relação a entrada. Enquanto que modelos que realizam uma busca para otimizar uma determinada condição, como DyS-Topsøe, HDy e SORD, além do tempo de predição do classificador, soma-se a complexidade de tempo dessa busca.

O algortimo proposto, SMM, apresenta o mesmo tempo de treinamento que grande parte dos modelos, porém, devido sua complexidade reduzida, possui a mesma complexidade de tempo de predição que algoritmos mais simples. E, mesmo assim, não apresentou diferença estatística para dois modelos considerados estado-da-arte: DyS-Topsøe e SORD. Ou seja, dos modelos avaliados em Hassan, Maletzke e Batista (2020), o SMM se mostrou um dos mais rápidos e performáticos.

3.1.2 *A comparative evaluation of quantification methods*

Schumacher, Strohmaier e Lemmerich (2021) realizam vários experimentos com algoritmos estado-da-arte, especialmente em *datasets* multi-classe. No trabalho, foram considerados 24 quantificadores, onde alguns foram extendidos para suportar naturalmente problemas multi-classe. É um dos poucos trabalhos que lidam com conjuntos de dados multi-classe, enquanto que outros não consideram esse tipo de problema (BELLA et al., 2010; MILLI et al., 2013) ou transformam o conjunto de dados em um problema binário (FORMAN, 2005; GONZáLEZ-CASTRO; ALAIZ; ALEGRE, 2013; HASSAN; MALETZKE; BATISTA, 2020).

Para a avaliação dos modelos, os autores aplicaram uma versão modificada do protocolo APP. Foram escolhidas distribuições de classes, tanto para o treino quanto para o teste, de forma relativamente arbitrária. Diferentemente do padrão do protocolo, as distribuições das prevalências não são igualmente espaçadas, ou seja, existe um viés nos valores escolhidos. Para o caso binário, as prevalências da classe positiva no treino e no teste foram, respectivamente:

$$treino_{+} = \{0.05, 0.1, 0.3, 0.5, 0.7, 0.9\}$$
$$teste_{+} = \{0, 0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$$

Para os conjuntos multi-classe, que abrangem 3, 4 e 5 classes distintas, a Figura 5 ilustra as distribuições escolhidas, tanto para treino quanto para teste.

Adicionalmente, eles também variaram o tamanho, mais especificamente a proporção, dos conjuntos de treino e teste, utilizando as seguintes divisões:

$$\{(0.1, 0.9), (0.3, 0.7), (0.5, 0.5), (0.7, 0.3)\}$$

Dada essa configuração, foram realizadas as combinações possíveis entre as distribuições de treino, distribuições de teste e proporções de tamanho para avaliar os modelos.

L	Training Distributions $P_{Train}(Y)$	Test Distributions $P_{Test}(Y)$
3	$(0.2, 0.5, 0.3)$, $(0.05, 0.8, 0.15)$, $(0.35, 0.3, 0.35)$	$(0.1, 0.7, 0.2)$, $(0.55, 0.1, 0.35)$, $(0.35, 0.55, 0.1)$, $(0.4, 0.25, 0.35)$, $(0., 0.05, 0.95)$
4	$(0.5, 0.3, 0.1, 0.1)$, $(0.7, 0.2, 0.1, 0.1)$, $(0.25, 0.25, 0.25, 0.25)$	$(0.65, 0.25, 0.05, 0.05)$, $(0.2, 0.25, 0.3, 0.25)$, $(0.45, 0.15, 0.2, 0.2)$, $(0.2, 0, 0, 0.8)$, $(0.3, 0.25, 0.35, 0.1)$
5	$(0.05, 0.2, 0.1, 0.2, 0.45)$, $(0.05, 0.1, 0.7, 0.1, 0.05)$, $(0.2, 0.2, 0.2, 0.2, 0.2)$	$(0.15, 0.1, 0.65, 0.1, 0)$, $(0.45, 0.1, 0.3, 0.05, 0.1)$, $(0.2, 0.25, 0.25, 0.1, 0.2)$, $(0.35, 0.05, 0.05, 0.05, 0.5)$, $(0.05, 0.25, 0.15, 0.15, 0.4)$

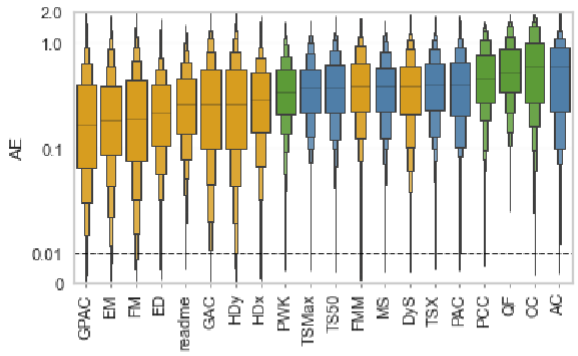
Figura 5 – Distribuições de classes escolhidas para o conjunto de treino e de teste. A coluna L representa a quantidade de classes distintas no conjunto.

Fonte: (SCHUMACHER; STROHMAIER; LEMMERICH, 2021)

Os resultados são reportados de diferentes perspectivas: (1) Erro Absoluto de cada quantificador entre todos os *datasets* avaliados, considerando todas as combinações possíveis descritas anteriormente; (2) Erro Absoluto de cada quantificador em função da diferença de prevalência entre o treino e o teste; (3) Erro Absoluto de cada quantificador em avaliações experimentais onde o conjunto de treino é relativamente menor que o de teste (separação do conjunto de dados em 10% para treino e 90% para teste). As conclusões dos autores a partir de cada perspectiva é descrita a seguir:

1. **Erro Absoluto geral**: para datasets binários, o DyS e o MS obtiveram a melhor performance geral tanto por meio do Erro Absoluto quanto do *Kullback-Leibler Divergence* Normalizado. Além disso, o ranqueamento médio dos dois modelos não apresenta diferença estatística. No contexto geral de conjuntos de dados multi-classe, existe uma inconstância maior na performance e ranqueamento dos modelos dependendo da métrica de erro escolhida, ou seja, o ranqueamento muda a depender da métrica. O modelo GPAC (FIRAT, 2016), uma versão multi-classe do PACC, obteve a melhor performance nesse contexto. Além disso, observou-se um aumento da magnitude dos erros em *datasets* multi-classe em comparação com *datasets* binários. A Figura 6 ilustra a distribuição dos Erros Absolutos de cada quantificador avaliado em conjuntos de dados multi-classe.

2. **Erro Absoluto em função da diferença de prevalência do treino e do teste**: em ambos os cenários, binário e multi-classe, os algoritmos que possuem as melhores performances no geral também apresentam boas performances em cenários com alta diferença de prevalência, ou seja, eles demonstram ser menos suscetíveis a essa mudança. Também é possível perceber uma piora sistemática de parte dos



(a) Distribution of absolute errors (AE)

Figura 6 – Distribuição do erro de cada modelo avaliado. O eixo y está na escala logarítimica.

Fonte: (SCHUMACHER; STROHMAIER; LEMMERICH, 2021)

algoritmos, principalmente dos modelos baseados em classificadores (descritos na Subseção2.2.1) conforme essa diferença aumenta.

3. **Erro Absoluto com treino menor que teste**: os experimentos mostram que o tamanho do treino, ou a quantidade de informação disponível no treino, influencia, no geral, a performance dos quantificadores: quanto menos dados, pior é a performance.

3.2 Quantificação Textual

Esta seção apresenta 5 trabalhos sobre quantificação no domínio textual. Na Subseção 3.2.1 é apresentado uma abordagem interativa, por meio da construção de uma ferramenta, para classificar e quantificar textos curtos. A Subseção 3.2.2 apresenta dois trabalhos sobre quantificação de sentimento de *tweets*, enquanto que a Subseção 3.2.3 apresenta um *survey* sobre análise de sentimento de *tweets*. Por fim, a Subseção 3.2.4 mostra a popularização da tarefa de quantificação à parte de classificação.

3.2.1 *Pragmatic text mining: minimizing human effort to quantify many issues in call logs*

Forman, Kirshenbaum e Suermondt (2006) apresentam uma ferramenta para analisar *logs* de chamadas de suportes técnicos e melhorar os serviços ao cliente. O objetivo principal é quantificar os problemas mais frequentes de cada produto. Para tanto, é utilizada uma base de anotações feitas pelo time de suporte ao conversar com os consumidores pelo telefone.

Embora existam múltiplos problemas, e, consequentemente, múltiplas classes, que um suporte possa pertencer, os autores modelam múltiplos problemas binários - ao invés de um único problema multi-classe. No domínio estudado, o número de categorias é altamente dinâmico e crescente, pois, muitas vezes, precisa ser descoberto nos próprios dados.

A tarefa foi divida em 3 partes: clusterização de dados crus; anotação manual dos exemplos baseada em algumas informações retiradas de seus *clusters*; e treinamento de um classificador e um quantificador para predição dos demais exemplos. Esse pipeline é feito iterativamente, de forma a retornar para o anotador o *feedback* tanto da clusterização quanto da rotulação. Uma vez que os modelos, classificador e quantificador, estão bons o suficiente, o editor pode exportá-los para uso *off-line*, ou seja, predizer outros conjuntos de instâncias fora da ferramenta.

Para a etapa de classificação, Forman, Kirshenbaum e Suermondt (2006) utilizaram um modelo baseado em *Naive Bayes*, principalmente para a etapa inicial que necessita de um tempo de resposta menor, e outro baseado em *Support Vector Machine*. Para a tarefa de quantificação, dois modelos foram utilizados: o *Adjusted Classify and Count* (2.2.1.2) e um *Mixture Model* (2.2.3.1).

Forman, Kirshenbaum e Suermondt (2006) apontam que na classificação, geralmente, é necessário um número grande de dados rotulados, principalmente em cenários com muitas categorias. No trabalho, afirmam que uma pequena quantidade de instâncias rotuladas pôde gerar um ganho maior com métodos de quantificação do que puramente com classificação - menos esforço com melhor performance. Entranto, não há compartilhamento dos resultados sobre a qualidade dos procedimentos adotados envolvendo, principalmente, o problema de quantificação atacado.

3.2.2 Análise de Sentimento de *Tweets*

Em Gao e Sebastiani (2016) e Moreo e Sebastiani (2022) são apresentados estudos sobre a análise de sentimento textual, onde o objetivo final é quantificar cada um dos sentimentos possíveis - positivo, neutro e negativo - de *tweets*. Os autores argumentam que muitas vezes a classificação da polaridade de *tweets* tem como objetivo a quantificação e não a classificação de *tweets* individuais. Ou seja, o foco não é discriminar o sentimento de cada *tweets*, e sim, analisar a polaridade de um conjunto todo.

Para a extração de *features*, basearam-se no método proposto em Kiritchenko, Zhu e Mohammad (2014), específico para o domínio de *tweets*. Esse pré-processamento inclui remoção de links e menções a outros usuários da plataforma, tokenização e tagueamento *part-of-speech* (*POS tagging*). Para a construção de uma representação vetorial dos textos, utilizam uma mescla de *Bag-Of-Words* com uma estratégia de *n-grams*, e uma série de *features* com base em análises sobre a presença ou ausência de elementos textuais

e/ou contagem da frequência desses elementos nos textos - como ponto de exclamação e interrogação, contextos negativos, entre outros.

Embora seja um problema de classificação multi-classe, ambos os trabalhos utilizam uma abordagem *One-Vs-All*, pois os modelos de quantificação utilizados são majoritariamente binários. Geralmente, essa modelagem tende a facilitar o processo experimental da tarefa de quantificação. É um dos poucos trabalhos que considera *datasets* multi-classe para atacar problemas de quantificação.

Moreo e Sebastiani (2022) argumentam que os experimentos realizados por Gao e Sebastiani (2016) não são confiáveis e apresentam novos resultados com base em uma configuração experimental diferente - utilizando o *Artificial Prevalence Protocol* (2.3). Em Moreo e Sebastiani (2022), os autores variam a prevalência das 3 classes de 0 a 1, com um passo de 0.05. Diferentemente do caso binário, que possui uma configuração experimental mais simples, é necessário garantir que a distribuição, formada por uma tripla de valores, some exatamente 1. Seguindo esse processo e a restrição imposta, 231 distribuições distintas foram utilizadas para avaliar os modelos. Para cada distribuição, foram coletadas 25 amostras aleatórias distintas de 100 documentos cada que possuem a distribuição de classe respectiva.

Em Moreo e Sebastiani (2022), os classificadores base utilizados nos modelos de quantificação, baseados em regressão logística e SVM, são otimizados visando a tarefa de quantificação. Para tal, foi utilizado um conjunto de validação, disjunto do conjunto de treino e de teste. A avaliação nesse conjunto é similar ao conjunto de teste, com a diferença de que foram 5 amostras de 100 documentos para cada distribuição e que para alguns conjuntos foi necessário utilizar amostragem com reposição devido a falta de instâncias.

Os modelos ACC, PACC e SLD - também conhecido como EMQ (GAO; SEBASTIANI, 2016; HASSAN; MALETZKE; BATISTA, 2020) - obtiveram consistentemente as melhores performances na maioria dos *datasets* avaliados.

Os autores reportam uma diferença evidente entre os dois trabalhos, que possuem métodos de avaliação diferentes. O SLD, que apresentou um dos piores resultados no primeiro (GAO; SEBASTIANI, 2016), mostrou possuir a melhor performance no segundo (MOREO; SEBASTIANI, 2022). Inversamente, o CC e PCC apresentam resultados ruins no segundo, diferentemente do primeiro trabalho. Os autores argumentam que essa mudança de comportamento é explicada pela diferença entre a distribuição de treino e de teste que os modelos foram avaliados. O método APP, utilizado no segundo trabalho, força diferenças bem discrepantes, varrendo diversos valores possíveis de prevalência.

Além disso, Moreo e Sebastiani (2022) reportam o erro absoluto em função da diferença entre as distribuições de classes do treino e do teste. A performance do CC, modelo utilizado como *baseline* em vários trabalhos, tende a diminuir conforme a diferença aumenta. Ou seja, quanto maior a diferença consequentemente maior o erro na predição. A Figura 7 ilustra essa relação. É notável que alguns modelos, como o SLD, PACC e ACC,

são menos suscetíveis a essa mudança.

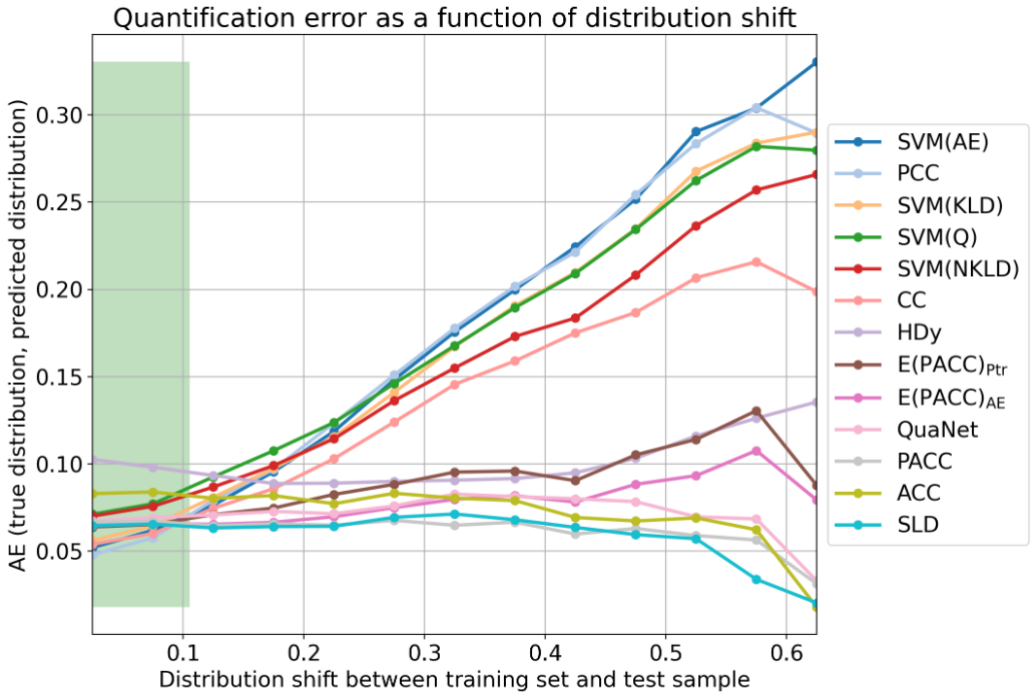


Figura 7 – Comportamento da performance de quantificadores conforme a diferença entre a prevalência de treino e teste aumenta.

Fonte: (MOREO; SEBASTIANI, 2022)

Por fim, os autores concluem que o segundo trabalho é mais completo que o primeiro, no sentido de que abrange muitos outros cenários de avaliação. Adicionalmente, reportam um entendimento maior do comportamento dos modelos previamente considerados bons - os primeiros resultados são verdadeiros somente em cenários com pouca diferença de distribuição entre o treino e o teste.

3.2.3 *Like it or not: A survey of twitter sentiment analysis methods*

Giachanou e Crestani (2016) apresentam um *survey* sobre análise de sentimento de *tweets*. O domínio em questão possui algumas características próprias que o diferencia das demais análises textuais. Os autores listam algumas dessas características que podem impactar na tarefa:

1. **Tamanho dos textos**: os *tweets* possuem um tamanho máximo de 140 caracteres;

2. **Tópico**: para a classificação de sentimento de um *tweet*, é importante considerar o tópico ou assunto que o texto aborda;

3. **Erros Gramaticais**: devido a natureza informal da plataforma e também pelo limite de caracteres imposto, uma grande parte dos *tweets* apresenta erros gramaticais, muitas vezes propositais, para simplificar o texto ou enfatizar uma ideia;

4. **Vocabulário Esparso**: uma grande parte dos termos utilizados no domínio apresenta uma frequência relativamente baixa (SAIF; HE; ALANI, 2012);

5. **Negação**: a dectecção de elementos negativos no texto é um grande desafio dentro da tarefa de classificação textual, uma vez que pode ocasionar em uma inversão total de sentimento;

6. ***Stopwords***: palavras altamente frequentes porém genéricas que possuem baixo valor no momento de categorizar textos em determinadas polaridades;

7. **Tokenização**: tarefa de separar o texto em pedaços, *tokens*, geralmente palavras. Entretanto há outras técnicas mais complexas que apresentam bons resultados no domínio de *tweets* (OWOPUTI et al., 2013);

8. **Multi-língua**: por ser uma plataforma global, existe uma varidade grande de línguas que podem ser encontradas no domínio. Muitas vezes um mesmo *tweet* pode conter mais de uma única língua. Essa característica cria a necessidade de modelos que suportem múltiplas línguas para solucionar o problema;

9. ***Multimodal***: além dos textos propriamente ditos, os *tweets* podem apresentar imagens e vídeos que podem ser utilizados para auxiliar na tarefa. Porém, considerar essas outras fontes de informação aumenta a complexidade da tarefa, uma vez que modelos de naturezas distintas são necessários para classificá-los.

No domínio dos conjuntos de dados textuais, há diversas técnicas densenvolvidas para pré-processar os textos de forma que sejam utilizáveis/interpretáveis por algoritmos. Giachanou e Crestani (2016) apresentam categorias de *features* mais comuns em trabalhos relacionados à análise de sentimento de *tweets*:

1. **Sintática**: provavelmente as mais comuns, estão nessa categoria as próprias palavras, *bigrams*, *n-grams*, a frequência de termos textuais e rótulos *part-of-speech*;

2. **Semântica**: palavras ou termos que possuem uma habilidade discriminativa muito forte para sentimento, geralmente anotadas manualmente. Termos negativos também são bem relevantes, pois podem inverter completamente o sentimento de uma frase ou sentença;

3. **Estilística**: características da escrita no contexto de *microblog*, como *emoticons*/*emojis*, abreviações, gírias e pontuações. A repetição excessiva de caracteres e a escrita em caixa alta (maiúsculo) do texto também entram nessa categoria;

4. **Específicas da Plataforma**: *hashtags* utilizadas, quantidade de *retweets*, respostas de outros usuários, menções à outras pessoas, quantidade de seguidores e URLs fazem parte desta categoria.

Os autores listam dois trabalhos (ESULI; SEBASTIANI, 2015; AMATI; BIANCHI; MARCONE, 2014) que atacam o problema de quantificação textual no contexto de *tweets*. Problema que tem se tornado relevante após ser incluído nas tarefas do *SemEval*-2016[1].

3.2.4 *SemEval* 2016

Nakov et al. (2019) apresentam os resultados obtidos na tarefa 4 do Workshkop Internacional focado em PLN *SemEval*[2] do ano de 2016. A tarefa 4 corresponde à análise de sentimento de *tweets* e engloba outras subtarefas. Especificamente nessa edição, são introduzidas subtarefas relacionadas à quantificação - subtarefas D e E. Para ambas as subtarefas, o objetivo é induzir um modelo para quantificar um conjunto de teste único e específico. Ou seja, uma configuração experimental simplificada e baseada no NPP.

Para a subtarefa D, quantificação binária, foram usados dois modelos *baselines*: (1) utilizar a prevalência do treino como predição; (2) a classe majoritária, no treino, recebe 1 enquanto que a minoritária recebe 0, sempre. A medida de avaliação escolhida foi uma versão suavizada da métrica *Kullback-Leibler Divergence* (KLD) (FORMAN, 2005). Todos os modelos propostos foram capazes de superar ambos os *baselines*. Apenas 5 dos 14 modelos participantes eram realmente quantificadores, enquanto que o modelo que obteve a melhor performance utilizou a estratégia mais simples de *Classify and Count*.

A subtarefa E, quantificação multi-classe, também contou com dois modelos *baselines* similares a da subtarefa descrita anteriormente, porém em suas versões que suportam múltiplas classes. Nessa subtarefa, foi adotada a métrica *Earth Mover's Distance* (EMD) (RUBNER; TOMASI; GUIBAS, 2000) para avaliação dos modelos propostos. Dos 10 modelos participantes, somente 3 eram quantificadores. Entretanto, diferentemente da subtarefa D, o modelo que obteve a melhor performance foi um algoritmo novo implementado especificamente para a quantificação multi-classe.

Nakov et al. (2019) propõem inserir na avaliação experimental das tarefas de quantificação o protocolo APP, onde vários conjuntos de teste com prevalências distintas são gerados artificialmente. Dessa forma, seria possível considerar a prevalência de classe do teste e a diversas diferenças de prevalência entre treino e teste que não necessariamente aparecem nos conjuntos de dados.

[1] https://semeval.github.io/
[2] https://semeval.github.io/

3.3 Quantificação Temporal

Esta seção contém trabalhos relacionados a quantificação ao longo do tempo. A Subseção 3.3.1 apresenta um estudo sobre o comportamento de posts com sentimento positivo ao longo do tempo e a Subseção 3.3.2 apresenta um trabalho sobre quantificação em *data streams*.

3.3.1 *Improving sentiment analysis accuracy with emoji embedding*

Em Liu et al. (2021), os autores apresentam o *Chinese emoji-embedding LSTM*, (*CEmo-LSTM*), uma rede neural para classificar o sentimento de *tweets* levando em consideração o uso de *emojis*. A arquitetura da rede neural conta com uma camada de *embeddings*, representação vetorial de um símbolo, para os *tokens* e também para os *emojis*.

Três experimentos principais são conduzidos: (1) comparação entre o uso somente do texto contra a adição dos *emojis*; (2) substituição dos *emojis* por palavras que remetem ao sentimento associado a eles (tristeza, felicidade, etc.); (3) treinamento dos classificadores somente com textos que possuem emojis alinhados com o texto, ou seja, não contradizem o sentimento expresso pelo texto. Em todos os experimentos, os autores comparam vários modelos clássicos utlizados em tarefas de análise de sentimento, juntamente com o *CEmo-LSTM*.

Liu et al. (2021) concluem que o uso de *emojis* melhora a acurácia dos classificadores de forma geral, mas treinar os modelos somente com instâncias onde os *emojis* reforçam o sentimento do texto é a melhor abordagem. Os *emojis* contraditórios ao texto tendem a diminuir a a acurácia dos modelos.

Além disso, os autores fazem um estudo de caso sobre *posts* da plataforma *Weibo* [3] dos residentes de Wuhan durante a pandemia de COVID-19. Para tanto, o *CEmo-LSTM* foi utilizado para classificar *posts* com sentimento positivo e negativo, agregados por dia, num período de quatro meses. A figura 8 ilustra a quantidade de *posts* positivos ao longo do tempo.

Apesar da quantidade absoluta reportada pelos autores não ser suficiente para concluir que os *posts* positivos realmente diminuiram após o surgimento do COVID-19, entre 23/01 e 30/01 (LIU et al., 2021), vale ressaltar o uso do método CC para a obtenção desses dados. Embora seja conhecido que esse método não apresenta bons resultados em diversos trabalhos (FORMAN, 2005; FORMAN, 2006; MALETZKE et al., 2019; MOREO; SEBASTIANI, 2022; SCHUMACHER; STROHMAIER; LEMMERICH, 2021), ele ainda é utilizado em contextos de quantificação, principalmente como *baseline*.

[3] www.weibo.com

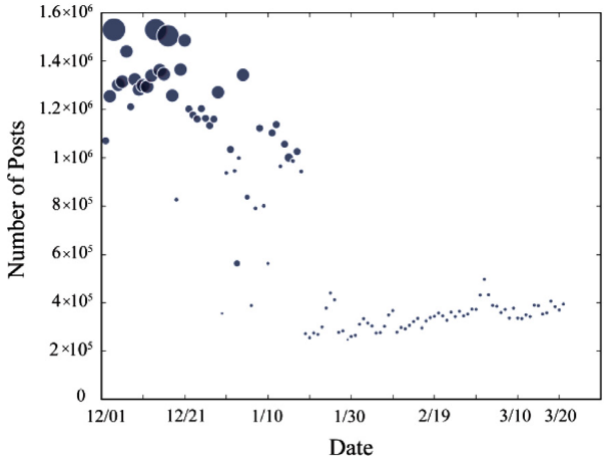


Figura 8 – Quantidade de *posts* positivos entre dezembro e março de 2020 na plataforma *Weibo*.

Fonte: (LIU et al., 2021)

3.3.2 *Quantification in data streams: Initial results*

Maletzke, Reis e Batista (2017) apresentam um estudo sobre quantificação em fluxo (ou *streaming*) de dados. Um fluxo é uma sequência ordenada de instâncias, geralmente associada à estrutura temporal dos dados. Para problemas de classificação e/ou quantificação, essas instâncias possuem um rótulo associado. Ou seja, podemos descrever um *streaming* de dados S como $S = [(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)]$.

Esse domínio possui características próprias e impõe certos desafios, não só pelo caráter ordenado do conjunto de instâncias, mas também por estar associado à problemas que apresentam mudança de conceito (GAMA et al., 2004). Essa mudança pode estar relacionada à variação na probabilidade *a priori* das classes conhecidas $P(y_i)$, na distribuição das instâncias $P(x_i|y_i)$ ou na probabilidade condicional das classes $P(y_i|x_i)$.

Outro desafio em cenários de fluxo de dados é a latência para se obter os rótulos reais de novos dados, ou seja, o tempo entre o surgimento/coleta do dado até a obtenção do rótulo verdadeiro (REIS et al., 2016). Em problemas reais, a latência não é baixa, e muitas vezes os rótulos reais nunca chegam a ser obtidos.

No trabalho, são descritos dois protocolos comuns em *data streaming*: (1) **estático**, em que o modelo é treinado somente uma vez, com os dados do início do fluxo; (2) **deslizante**, em que o modelo é retreinado periodicamente a cada 300 eventos, com os dados mais atuais do fluxo. Os autores propõem um algoritmo para solucionar o problema de mudança de conceito (*concept drift*) conforme os eventos ocorrem. Entretanto, os resultados estão agregados somente a nível de modelo, e não é possível saber o comportamento da performance de cada quantificador ao longo da cadeia de eventos. Ou seja, não fica claro se a qualidade dos modelos melhorou, se manteve ou piorou conforme novas instâncias eram preditas com o decorrer do tempo.

Os autores proprõem o *Data Stream Quantification by Score Inspection* (SQSI) que

utiliza um teste estatístico para detectar mudanças de conceito. Dado um novo conjunto de eventos a ser predito pelo modelo, o algoritmo verifica, por meio do teste estatístico *Kolmogorov-Smirnov*, se os *scores* obtidos das instâncias do treino pertencem à mesma distribuição dos *scores* do treino - obtidos via validação cruzada. Esses *scores* são gerados por um classificador base presente no algoritmo. Se a hipótese nula não pode ser rejeitada, a quantificação é feita para o conjunto de eventos corrente. Entretanto, caso contrário, ou seja, os *scores* de treino e teste comparados não pertencem a mesma distribuição, então uma transformação é aplicada no conjunto de treino, o classificador é atualizado e novos *scores* são gerados.

Novamente o teste estatístico é aplicado, e se novamente a hipótese nula é rejeitada, os rótulos das instâncias são requisitados para anotação manual e o classificador base é atualizado. Caso contrário, o conjunto de eventos é quantificado.

Para entender a performance do método proposto, Maletzke, Reis e Batista (2017) comparam o SQSI com duas abordagens descritas anteriormente, consideradas como pior e melhor cenário: estática e deslizante, respectivamente. A abordagem estática ignora as possíveis mudanças de conceitos do conjunto de dados e não atualiza o quantificador ao longo do tempo, que é treinado somente uma única vez com o conjunto inicial de instâncias. Por outro lado, a deslizante, por atualizar o modelo frequentemente, consegue incorporar as mudanças ao longo do tempo.

Os resultados reportados estão agregados apenas a nível de *dataset*, o que não nos permite saber qual o comportamento do desempenho dos modelos ao longo do tempo. Outro ponto é o uso restrito de quantificadores de natureza corretiva - CC, PCC e ACC.

Por fim, os autores reportam que o algoritmo SQSI proposto não possui diferença estatística em relação à abordagem deslizante, considerada como o melhor cenário (sem latência). Esse resultado é valido para todos os 3 quantificadores avaliados no trabalho.

Capítulo 4

Quantificação Temporal

Neste capítulo serão descritos o problema a ser resolvido por este trabalho e a configuração experimental adotada.

4.1 Descrição do Problema

O problema atacado por este trabalho é a quantificação ao longo do tempo de *batches* de textos. Mais especificamente, o objetivo é analisar como os modelos presentes neste trabalho se comportam conforme novos dados surgem para serem quantificados. É importante salientar que o surgimento desses dados respeita uma ordem cronológica, e os métodos propostos incorporam essa restrição.

Para ter um entendimento complemento sobre a análise deste trabalho, quatro protocolos experimentais foram adotados:

1. **APP**: protocolo comumente utilizado para avaliar modelos de quantifição (Seção 2.3);

2. **Estático**: método que treina o quantificador apenas uma vez e o utiliza para futuras predições;

3. **Dinâmico**: método que atualiza o quantificador ao longo do tempo, na tentativa de incorporar possíveis mudança nos dados;

4. ***Forecasting***: modelar o problema como uma tarefa de *forecasting* da prevalência das classes conhecidas.

Os resultados obtidos por meio do APP serão usados como forma de consulta em relação o desempenho geral dos quantificadores. Esse protocolo permite entender quais modelos são mais suscetíveis a mudanças de prevalência - presentes ou não nos dados reais. Neste primeiro experimento, o aspecto temporal dos dados é ignorado, ou seja, a ordem cronológica das instâncias, ou *batches*, não é levada em conta durante o procedimento.

Os outros 3 experimentos serão utilizados para entender o problema da perspectiva temporal. A abordagem estática é considerada, neste trabalho, como a configuração padrão para esse tipo de problema. Por outro lado, a abordagem dinâmica serve como um limite superior de desempenho, uma vez que seria o caso ideal onde os rótulos dos dados recentemente gerados estão prontamente disponíveis para consulta. Por último, a abordagem de *forecasting* é uma tentativa de simplificar o problema de quantificação temporal para uma modelagem de série temporal.

4.2 Materiais

O domínio desta pesquisa engloba dados textuais, pois esse contexto possui uma disponibilidade maior de meta-dados sobre o que consideramos instâncias, no caso, textos. Além disso, este tipo de dados é comum em aplicações que, naturalmente, dependem da tarefa de quantificação, como a análise de sentimentos sobre um produto em plataforma de e-commerce ou tópico em mídia social. Para considerar a cronologia dos dados, se faz necessária a informação de data de criação de cada texto disponível. Em outros domínios, essa informação não está presente, ou não se aplica ao contexto do problema.

A Tabela 2 descreve algumas informações sobre os conjuntos de dados utilizados nesse trabalho: nome, nível de agregação, número de instâncias, média de palavras e quantidade de classes.

Tabela 2 – Informações dos conjuntos de dados usados nos experimentos.

conjuntos de dados	Agregação	Instâncias	Média de Palavras	Classes
amazon	mês	44955	30	5
rating	dia	129098	23	5
recom	dia	129080	23	2
corona	semana	44955	30	5
olist	mês	40977	11	5
topic	mês	6997	27	6

O termo *batch* será usado para se referir a um conjunto de instâncias pertencentes a um determinado tempo t existente no conjunto de dados. O tempo t será definido pelo nível de agregação do conjunto de dados, podendo ser dia, semana ou mês. Ou seja, instâncias que foram criadas na mesma data ou janela de tempo (respeitando a agregação) pertencem ao mesmo *batch*. Além disso, a notação t_i indica a ordenação do *batch* no tempo - posicionando, por exemplo, t_2 depois de t_1 e antes de t_3. A Figura 9 ilustra essas definições juntamente com a prevalência das classes de cada *batch*.

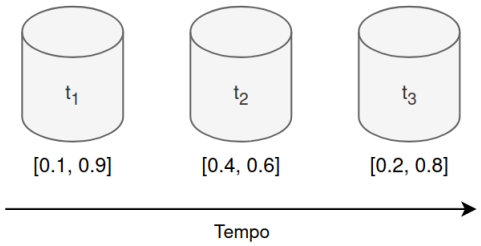


Figura 9 – *Batches* e suas respectivas distribuições de classes ordenados no tempo.

Para o pré-processamento dos dados, foi aplicada a técnica de *Bag-Of-Words* considerando somente as 10.000 palavras mais recorrentes no *corpus* de treinamento. Nesta abordagem, os textos são convertidos em uma representação vetorial, onde cada posição representa a frequência de uma palavra respectiva no texto. Adicionalmente, os textos foram formatados para *lowercase*.

4.3 Métodos

4.3.1 Modelos

O modelo baseado em *Regressão Logística* foi escolhido como classificador base para os modelos de quantificação por ser simples e obter bons resultados em problemas de classificação textual. Dessa forma, a comparação dos quantificadores se mantém mais simples, uma vez que todos possuem o mesmo classificador base.

Para reduzir o conjunto de quantificadores avaliados, optou-se por realizar alguns experimentos preliminares a fim de utilizar somente os que obtiveram os melhores resultados. A Tabela 3 apresenta os resultados desses experimentos através do protocolo APP. Os modelos selecionados foram: CC, ACC, PCC, PACC, DyS-Topsøe, SORD e SMM.

Tabela 3 – Erro Absoluto dos quantificadores por conjuntos de dados.

Quantificadores	Datasets						
	toxic-en	olid	sms-spam	restaurant-sent	fake-text	toxic-br	computer-sent
DyS-Topsøe	**0.0182**	**0.0371**	**0.0156**	**0.0458**	0.2441	0.1104	0.1335
SORD	0.0262	0.0434	0.0177	0.0542	**0.2366**	**0.1071**	0.1370
SMM	0.0221	0.0397	0.0199	0.0549	0.2500	0.1143	**0.1318**
MAX	0.0430	0.0506	0.0173	0.0567	0.2416	0.1321	0.149
X	0.0527	0.0507	0.0175	0.0606	0.2498	0.1351	0.1515
PACC	0.0208	0.0387	0.0177	0.0585	0.3182	0.1303	0.1673
HDy	0.0193	0.0439	0.0257	0.0730	0.2772	0.1975	0.1668
MedianSweep	0.0282	0.0429	0.0199	0.0647	0.3619	0.1910	0.1937
ACC	0.0341	0.0495	0.0283	0.0856	0.3740	0.1798	0.2147
PCC	0.2108	0.2407	0.0698	0.1945	0.3850	0.2985	0.2573
CC	0.1875	0.2549	0.1018	0.2359	0.4386	0.3300	0.3042
T50	0.2838	0.2987	0.1053	0.3120	0.4946	0.3834	0.3357

Os modelos CC, PCC, ACC e PACC foram escolhidos devido à sua natureza simples e de fácil implementação. No caso do CC, é comum utilizá-lo como referência principal em

questão de desempenho, uma vez que é considerado o caso base para a quantificação. Os modelos PCC e PACC, como descrito por Bella et al. (2010), foram calibrados por meio de uma regressão isotônica (NICULESCU-MIZIL; CARUANA, 2005).

Os modelos DyS-Topsøe, SORD e SMM obtiveram os menores valores de erro absoluto, por isso foram escolhidos para os experimentos deste trabalho. Em relação à implementação do DyS e do HDy usados nesta pesqusa, Hassan, Maletzke e Batista (2021) sugerem adotar uma busca ternária para encontrar o α que minimiza a equação de ambos os modelos (Equações 8 e 9) em vez de uma busca linear em todos os valores possíveis.

Além disso, Hassan, Maletzke e Batista (2021) levantam uma questão sobre a avaliação dos quantificadores por meio do APP. No protocolo em questão, existe um viés de configuração que pode ser observado pela distribuição uniforme das prevalências das classes durante a construção dos subconjuntos de teste. Essa uniformização pode causar uma distorção na interpretação dos resultados, pois tende a favorecer o caso médio da quantificação - nessa situação, um quantificador binário que sempre retorna 50% (ou 0.5) para ambas as classes pode ser beneficiado por essa configuração experimental.

Para evitar conclusões distorcidas sobre os modelos, dois quantificadores baseados no *Lazy Quantifier* foram adicionados aos experimentos deste trabalho: *Lazy Quantifier Balanced* (referido também como *LQ Balanced*), que retorna uma distribuição balanceada entre todas as classes conhecidas; e o *Lazy Quantifier Train* (*LQ Train*) que retorna sempre a prevalência presente no conjunto de treino. Com isso, é possível comparar se os outros quantificadores são melhores que, pelo menos, o caso médio do protocolo e entender o quão diferente os *batches* de teste são do conjunto de treinamento.

Como muitos quantificadores presentes neste trabalho são binários, utilizou-se a técnica de *One-Vs-All*, inclusive em conjuntos de dados binários (como é o caso do conjunto de dados *recom*) e modelos que suportam quantificação multi-classe - CC e PCC naturalmente, e ACC e PACC como apontado por Firat (2016) - a fim de manter o padrão nos experimentos conduzidos. Mesmo no caso binário, dois modelos são construídos para a realizar a predição, um para a classe positiva e outro para a classe negativa. Essa padronização é importante pois a técnica *One-Vs-All* não considera a dependência de uma classe sobre as outras no momento da predição, embora necessite de uma normalização ao final.

Adicionalmente, após a validação cruzada para obter artefatos como as taxas de *tpr* e *fpr* ou as distribuições de *scores* positivos e negativos para o treinamento dos quantificadores, os classificadores foram retreinados com todo o conjunto de treinamento para a etapa de teste.

Para a tarefa de regressão, 3 modelos foram avaliados neste trabalho: Regressão Linear, SVR e um *Multilayer Perceptron* (MLP).

4.3.2 *Artificial-Prevalence Protocol*

Duas formas de avaliar tarefas de quantificação são discutidos por Hassan, Maletzke e Batista (2021): ***artificial-prevalence protocol*** (APP) e ***natural-prevalence protocol*** (NPP). Neste trabalho, ambos os protocolos serão utilizados: o APP por ser um método amplamente utilizado na literatura e o NPP devido à configuração específica dos conjuntos de dados obtidos. Dessa maneira, é possível comparar como os quantificadores avaliados se comportam em cada um dos protocolos.

Como descrito na Seção 2.3, o APP consiste em dividir o conjunto de dados rotulados em treinamento e teste - como um problema de classificação usual. Entretanto, no momento da avaliação do conjunto de teste, há uma diferença crucial, e necessária, para se avaliar quantificadores: diversos subconjuntos são gerados a partir do conjunto de teste, variando-se a prevalência das classes. Desse modo, avalia-se os modelos em cada subconjunto gerado.

Há algumas características importantes sobre a forma como o protocolo foi implementado nesta pesquisa: (1) dividiu-se o conjunto de dados de forma estratificada em 5 *folds* de mesmo tamanho; (2) 5 avaliações foram conduzidas utilizando 4 *folds* como conjunto de treinamento e 1 *fold* como teste; (3) o conjunto de treinamento não foi balanceado para melhor aproveitar a quantidade de instâncias; (4) gerou-se as distribuições de teste de forma uniforme por meio de um passo único a depender da quantidade de classes; (5) para cada distribuição de teste, 10 subconjuntos distintos foram criados via amostragem sem repetição. A Figura 10 ilustra a configuração experimental adotada neste trabalho.

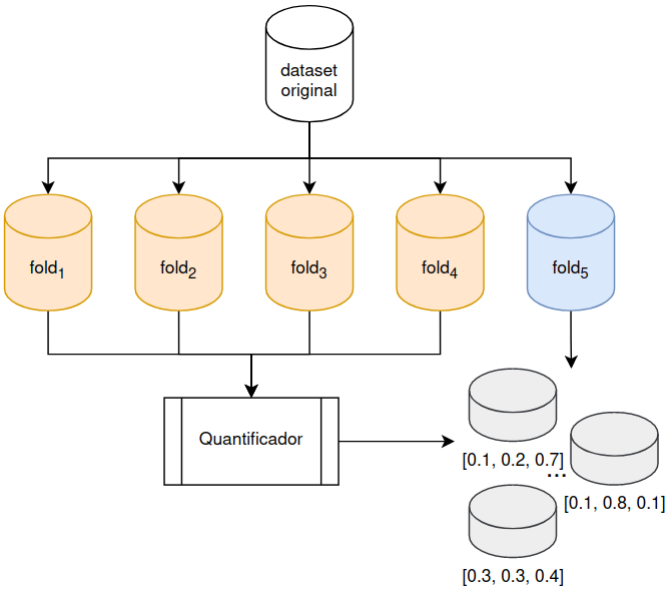


Figura 10 – Ilustração da implementação do APP de um conjunto com 3 classes. Esse procedimento é repetido 5 vezes, alternando os *folds* do treinamento e do teste.

As características (1), (2) e (5) são maneiras de adicionar uma variação a nível de ins-

tância tanto para o treinamento quanto para o teste, de forma a capturar a generalização dos algoritmos. Devido à característica (3), se fez necessária a adição de duas variantes do modelo *Lazy Quantifier*, o balanceado e o do treinamento (discutidos na Subseção 4.3.1).

Muitos trabalhos sobre quantificação utilizam apenas conjuntos de dados binários. Essa escolha facilita o processo de avaliação do APP, uma vez que é possível iterar linearmente entre as possíveis distribuições da classe positiva, geralmente o intervalo $[0, 1]$, e calcular a classe negativa como seu complemento. Entretanto, para conjuntos de dados multi-classe, o processo de gerar uniformemente distribuições distintas se torna um desafio, pois a quantidade cresce exponencialmente e já não conseguimos inferir as distribuições tão facilmente como no caso binário. De forma a tornar o processo factível, adotou-se um passo arbitrário entre as distribuições baseado na quantidade de classes conhecidas, de forma a diminuir a quantidade de distribuições distintas - característica (4). A Tabela 4 apresenta as classes, o passo e a quantidade de prevalências de teste de cada conjunto de dados.

Tabela 4 – Quantidade de prevalências de teste de cada conjunto de dados.

conjunto de dados	Classes	Passo	Prevalências de Teste
amazon	5	0.09	1160
rating	5	0.09	1160
recom	2	0.01	101
corona	5	0.09	1160
olist	5	0.09	1160
topic	6	0.15	407

4.3.3 Quantificação Temporal

Nesta seção serão descritas as abordagens de quantificação que consideram o aspecto temporal dos dados, ou seja, incorporam no momento da avaliação a estrutura cronológica dos dados. São elas: 1) **Estática**: abordagem onde os modelos são treinados somente uma única vez; 2) **Dinâmica**: abordagem onde os modelos são retreinados periodicamente, utilizando-se uma janela de dados passados. Vale ressaltar que datas sem dados não foram consideradas, uma vez que para um problema de quantificação esse cenário é trivial.

4.3.3.1 Estática

Nesta abordagem, os dados são divididos entre treino e teste a partir de uma determinada data de referência. Para os experimentos conduzidos neste trabalho, foi utilizada a data mais antiga que contemplasse pelo menos **30%** das instâncias rotuladas. A Figura 11 ilustra essa divisão. Ou seja, os *batches* que pertencem a uma data anterior à de referência ou igual serão parte do treino, enquanto o restante será considerado teste.

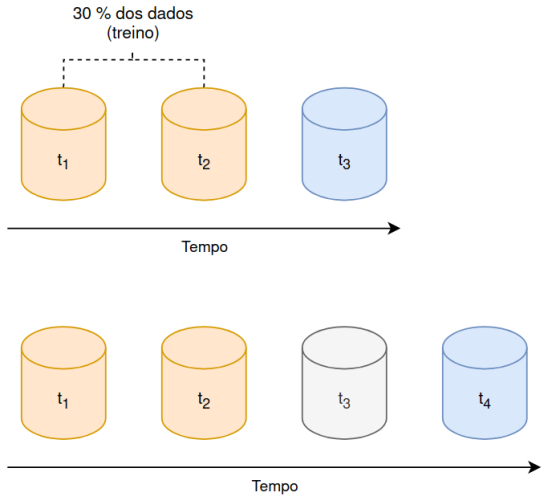


Figura 11 – Procedimento da abordagem estática. Os *batches* t_1 e t_2 são utilizados para induzir o quantificador. Uma vez predito t_3, ele é ignorado para predições futuras.

Os batches de treino são considerados como um único conjunto no momento do treinamento dos modelos, sem distinção temporal entre eles. Contudo, os batches de teste são avaliados **individualmente**, respeitando assim a divisão temporal entre eles.

4.3.3.2 Dinâmica

Nesta outra abordagem, a divisão entre treino e teste é igual à descrita anteriormente, ou seja, **30%** dos dados iniciais ficam *disponíveis* para o treinamento. Entretanto, após a avaliação de um *batch* de teste, ele é incorporado no conjunto de treinamento, e os quantificadores são retreinados. Esse procedimento é realizado respeitando a cronologia dos *batches* de teste. Há, então, dois parâmetros durante a avaliação dos modelos nessa abordagem: 1) **a janela de *batches* passados** que serão utilizados para retreinar os modelos; 2) **a periodicidade** que os modelos são retreinados. A Figura 12 a seguir ilustra esse procedimento:

Para reduzir as combinações possíveis entre os parâmetros e a complexidade do experimento, a periodicidade foi fixada em 1, a cada 1 *batch* de teste, e a janela foi fixada no tamanho máximo, ou seja, retreino histórico com todos os dados passados.

A técnica de retreino histórico será utilizada como *topline* - simulando uma situação ótima em que os rótulos reais das instâncias de teste estarão disponíveis num curto espaço de tempo. Essa abordagem não é real na prática, uma vez que, se rótulos de teste forem disponibilizados em tempo hábil, o problema de quantificação deixaria de existir. Porém, a abordagem serve como um limite superior de desempenho, visto que os modelos, a cada nova predição, são retreinados com mais dados - quanto maior o conjunto de treinamento tanto maior será a qualidade da predição dos quantificadores.

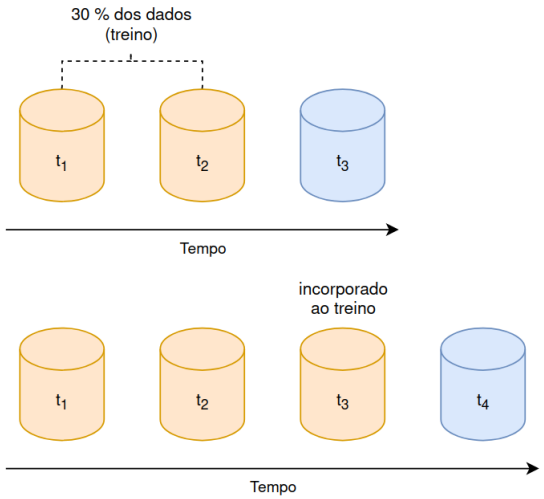


Figura 12 – Abordagem dinâmica, onde, para a predição de t_4, o *batch* t_3 predito anteriormente é incorporado ao treinamento dos modelos, que são atualizados.

4.3.4 Simplificação para *Forecasting*

Dada a estrutura cronológica dos dados, é natural enxergar o problema como uma série temporal. Nesta abordagem, ao invés de modelos de quantificação, o foco passa a ser induzir regressores para predizer a prevalência de cada classe no próximo *batch*. Essa abordagem é mais simples pelo fato de possuir somente um único modelo, ao contrário da quantificação - todos os quantificadores usados nesta pesquisa necessitam de um classificador base, ou seja, são necessários dois modelos para treinar e predizer. Além disso, as instâncias se tornam um conjunto ordenado da prevalência de cada classe em uma determinada janela de tamanho n - o que reduz a quantidade de instâncias disponíveis. Pode ser considerada como um *baseline*, uma vez que assume que a prevalência de um *batch* depende somente de *batches* anteriores. A Figura 13 a seguir ilustra essa modelagem:

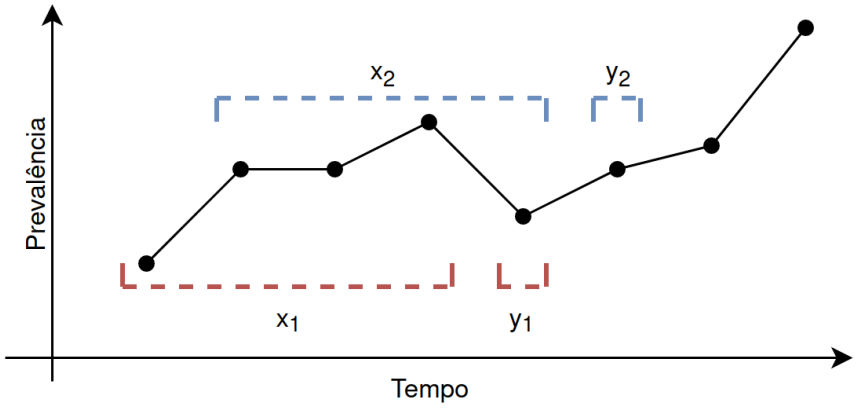


Figura 13 – Exemplo da modelagem do problema por meio de uma série temporal com uma janela de tamanho 4.

Em problemas de *forecasting* é comum variar a janela de valores que serão considerados como *features* para o modelo. Neste trabalho, somente a janela de tamanho 3 foi utilizada

devido ao tamanho dos conjuntos de dados após a transformação necessária para séries temporais. Alguns conjuntos de dados não possuírem mais de 3 *batches* que incorporassem os 30% de dados iniciais no treino. Mesmo que em alguns cenários essa quantidade de dados de treino seja insuficiente, a comparação desta abordagem se mantém coerente com as demais, estática e dinâmica.

Assim como na quantificação, o problema foi divido em duas técnicas de predição diferentes: 1) **Propagativa**: as predições individuais de *batches* anteriores são reutilizadas em futuras predições; 2) **Real**: as *features* das instâncias *sempre* possuem os valores reais das prevalências. Novamente, a predição propagativa se baseia num cenário real, onde os rótulos das instâncias não estão prontamente disponíveis e é necessário reutilizar as predições do modelo, enquanto que a propagação real considera que os rótulos serão fornecidos em tempo ábil para a predição do próximo *batch*.

A Figura 13 apresenta a abordagem (2) Real, uma vez que na instância x_2 somente os valores reais da série são utilizados. A Figura 14 ilustra a abordagem (1) Propagativa. Note que, nesse caso, a instância x_2 possui como feature a predição de x_1, ou seja, $\hat{y}_1$.

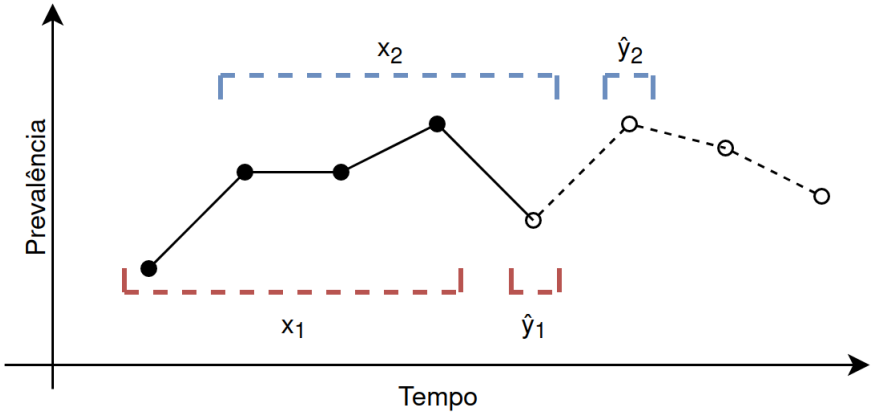


Figura 14 – Exemplo da abordagem propagativa. Os pontos preenchidos são valores reais da série enquanto que os **não** preenchidos representam as predições do modelo ao longo do tempo.

Capítulo 5

Resultados

Este capítulo apresenta os resultados dos experimentos conduzidos nesta pesquisa. Está divido em 3 seções:

- ***Datasets* Originais**: descreve a performance dos modelos de quantificação através das técnicas propostas neste trabalho utilizando os *datasets* coletados (sem nenhuma modificação);
- ***Datasets* Artificiais**: idêntico a seção anterior, porém utiliza uma versão modificada (artificialmente criada) dos *datasets* - indução de uma mudança de prevalência maior ao longo do tempo do que suas versões originais;
- **Forecasting**: apresenta os resultados obtidos quando o problema de quantificação temporal é modelado em um problema de regressão, mais especificamente de *forecasting*.

5.1 Datasets Originais

Nesta seção, estão dispostos os resultados dos experimentos com o protocolo APP na Subseção 5.1.1 e com a abordagem estática na Subseção 5.1.2.

5.1.1 *Artificial Prevalence Protocol*

Utilizando a implementação descrita na Seção 4.3.2, mediu-se a performance de cada quantificador nos *datasets* escolhidos. A Tabela 5 apresenta o Erro Absoluto Médio de cada quantificador entre todos os *datasets*.

Tabela 5 – Erro Absoluto Médio (EAM) de cada quantificador.

Quantificador	EAM
SORD	**0.107738**
DyS-Topsøe	0.110074
SMM	0.110815
ACC	0.114902
PACC	0.115672
PCC	0.154603
CC	0.156495
LQ Balanced	0.162549
LQ Train	0.195692

Os modelos baseados no *Lazy Quantifier* (*LQ Balanced* e *LQ Train*) apresentam os piores resultados. Ambos os resultados são esperados devido à natureza simples dos modelos. Entretanto, vale ressaltar que os quantificadores acima do *LQ Balanced* na tabela podem ser considerados bons, uma vez que superam sua performance em um cenário que favorece sua predição - como descrito na Seção 2.3.

A Figura 15 apresenta o ranqueamento médio dos quantificadores avaliados utilizando o protocolo APP. Embora muitos quantificadores tenham apresentado uma performance muito similar, tanto em termos de EAM quanto de ranqueamento médio, não foi observada nenhuma diferença estatística entre os pares de modelos avalidados através do teste de *Wilcoxon*. Uma das variáveis que impactou os testes é a quantidade de conjuntos de testes gerados durante o protocolo - especialmente para os *datasets* multi-classe.

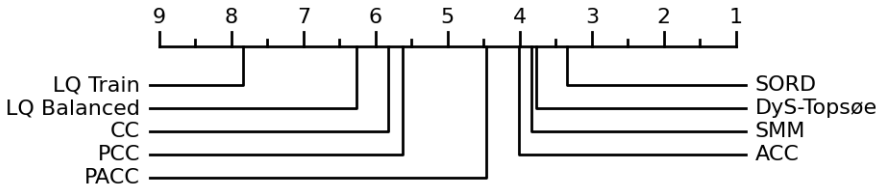


Figura 15 – Ranqueamento médio dos quantificadores.

No total, os modelos foram avaliados em 256.140 conjuntos de testes distintos. Segundo o teste de *Wilcoxon*, descrito na Equação 18, para $\alpha = 0.05$, a hipótese nula pode ser rejeitada caso z seja menor do que -1.96 - hipótese de que ambos os modelos avaliados possuem a mesma performance. Em muitos pares avaliados, como o par SMM e DyS-Topsøe, o fator N^2, onde N é a quantidade de conjuntos de teste, superou o valor de T bruscamente, o que consequentemente tornou os testes estatísticos sensíveis.

A Tabela 6 apresenta o resultado dos quantificadores detalhado por *dataset*. A coluna mais à direita traz a média entre os *datasets*. Os valores em negrito indicam que o quantificador da linha possui o menor erro da coluna.

Vale ressaltar que, especificamente para o *dataset corona*, o algoritmo que obteve melhor resultado foi o CC. Entretanto, no resto dos conjuntos de dados esse comportamento não se repetiu.

Tabela 6 – Erro Absoluto Médio de cada quantificador nos *datasets* avaliados.

Quantificador	amazon	rating	recom	corona	olist	topic	Média
SORD	0.147372	0.098442	0.002532	0.096575	**0.112980**	**0.064049**	**0.086992**
DyS-Topsøe	**0.140806**	0.103598	**0.002000**	0.100326	0.115074	0.068991	0.088466
SMM	0.145527	0.100337	0.003725	0.104791	0.113103	0.067023	0.089085
PACC	0.144713	0.105521	0.003168	0.111260	0.117747	0.070931	0.092223
ACC	0.169380	**0.086186**	0.004342	0.104504	0.120317	0.079540	0.094045
CC	0.226524	0.146058	0.087635	**0.084509**	0.201160	0.106733	0.142103
PCC	0.226716	0.141304	0.157422	0.118919	0.156083	0.113470	0.152319
LQ Balanced	0.168093	0.161114	0.252484	0.161223	0.160655	0.163387	0.177826
LQ Train	0.242330	0.177239	0.307876	0.165776	0.194865	0.190386	0.213079

5.1.2 Estático

Esta subseção apresenta os resultados da abordagem estática (4.3.3.1) para os *datasets* originais. A Tabela 7 contém o Erro Absoluto Médio dos quantificadores entre todos os *datasets* avaliados. O quantificador PCC demonstrou o menor erro, seguido do *LQ Train*, que possui uma abordagem simples de predição. Paralelamente, o *LQ Balanced* obteve o pior EAM do conjunto de modelos. Esses resultados indicam que os conjuntos de teste não estão balanceados porém possuem uma distribuição próxima a do treino.

Tabela 7 – Erro Absoluto Médio (EAM) de cada quantificador.

quantificador	EAM
PCC	**0.012275**
LQ Train	0.027765
DyS-Topsøe	0.040049
SORD	0.040685
SMM	0.043676
PACC	0.047589
CC	0.051169
ACC	0.051378
LQ Balanced	0.152933

Adicionalmente, a Figura 16 contém um *boxplot* do EAM de cada quantificador. O intervalo dos quartis do PCC são consideravelmente menores quando comparados com outros quantificadores, o que demonstra uma maior consistência do modelo entre os diferentes *batches* de teste. Ou seja, o erro do PCC é baixo independentemente do *batch* ou *dataset*.

A Figura 17 ilustra o ranqueamento médio dos quantificadores avaliados. Novamente, o PCC obteve o melhor *ranking* médio seguido do *LQ Train*. Os quantificadores baseados em *matching* de distribuição não apresentaram diferença estatística significativa entre si. Os resultados da Figura 17 colaboram com a Tabela 7, pois a ordenação de ambas perspectivas, erro e ranqueamento médio, se mantém quase que inalterada.

Entretanto, esses resultados vão de encontro aos resultados obtidos através do protocolo APP, descritos na Subseção 5.1.1. O quantificador PCC foi o melhor avaliado em termos de Erro Absoluto Médio e *ranking*, enquanto que o protocolo demonstrou que o quantificador SORD foi estatisticamente o melhor. Dado essa discrepância de resultados

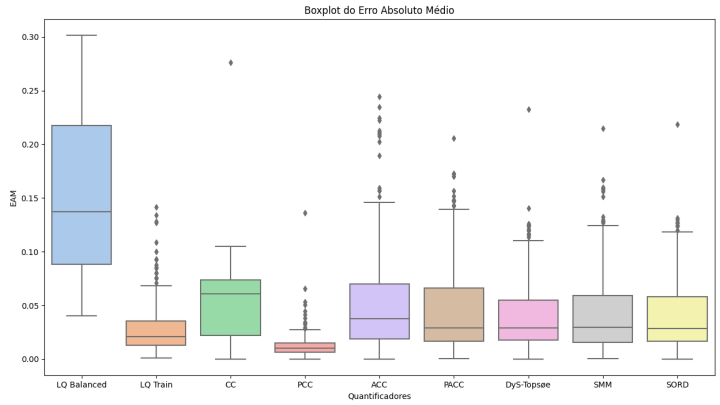


Figura 16 – *Boxplot* do EAM dos quantificadores avaliados.

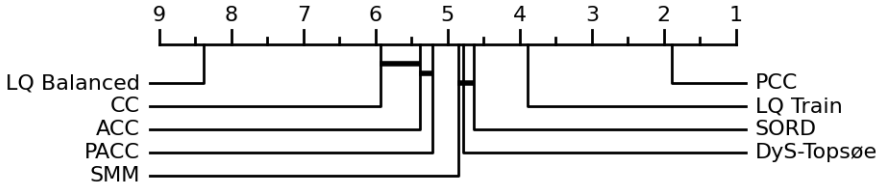


Figura 17 – Ranqueamento médio dos quantificadores. Os modelos ligados por uma linha horizontal não apresentam diferença estatística significativa entre si (teste de *Wilcoxon*).

em abordagens diferentes, foi feita uma análise em relação as prevalências de treino e as predições dos modelos em cada *dataset*, uma vez que elas são naturalmente definidas pelos dados.

A Figura 18 ilustra a relação do EAM com a diferença de prevalência entre o treino e a predição - *shift*. A cor dos pontos indica a qual *dataset* eles pertencem. Os modelos *LQ Train* e *LQ Balanced* possuem predições e a prevalência de seus conjuntos de treino sempre constantes nessa abordagem estática. Os pontos de ambos os modelos, portanto, recaem verticalmente sobre a diferença também constante.

Exceto pelos quantificadores *LQ Train* e o PCC, todos os outros quantificadores apresentam uma tendência de piorar conforme a diferença de prevalência entre treino e predição aumenta. Ou seja, se a predição é parecida com a prevalência do treino, então o erro é menor. Um caso extremo é justamente o comportamento do quantificador PCC, onde as predições possuem, em sua maioria, uma diferença menor que 0.05 para o treino e erros muito baixos. Para o quantificador *LQ Balanced*, essa tendência é observada entre os *datasets*, pois, num mesmo *dataset*, a diferença (eixo horizontal) é constante.

A Tabela 8 mostra as correlações de *Pearson* entre os dois eixos apresentados na Figura 18, *shift* e EAM, de cada quantificador. A correlação do quantificador *LQ Train* é indefinida uma vez que a diferença entre sua predição e a prevalência de treino é constante, independente do *dataset*. Exceto pelos modelos PCC e *LQ Train*, todos os outros apresentam um valor acima de 0.75 - o que indica uma correlação positiva forte.

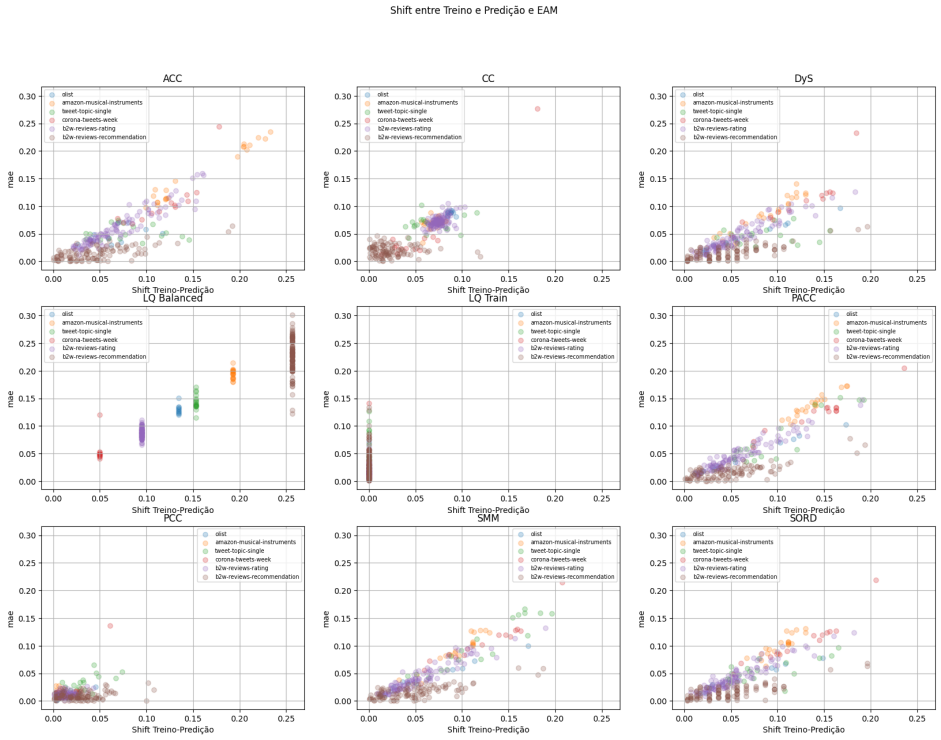


Figura 18 – *Shift* entre o treino e a predição e o Erro Absoluto Médio da predição.

Tabela 8 – Correlação entre EAM e a diferença de prevalência do treino e a predição de cada quantificador entre todos os datasets.

quantificador	correlação
LQ Balanced	**0.948813**
PACC	0.874777
ACC	0.858949
SMM	0.851312
CC	0.820436
SORD	0.808690
DyS-Topsøe	0.784339
PCC	0.298388
LQ Train	Indefinido

A Figura 19 ilustra o comportamento do *dataset rating* - os outros *datasets* estão dispostos no apêndice A.1. A reta vertical demarca até que data foi considerado conjunto de treino (incluso). É evidente que as prevalências possuem uma baixa variação ao longo do tempo e que o teste é quase uma cópia do treino. Outra característica notável é a de que a ordem das classes se mantém praticamente inalterada (ordem da classe com a maior prevalência para a classe com a menor prevalência) entre os conjuntos de teste.

Ao analisar mais profundamente o comportamento das prevalências de cada *dataset*, é possível perceber que o *shift* entre o treino e o teste ao longo do tempo é relativamente baixo, como mostra a Figura 20. Todos os pontos recaem sobre o intervalo $[0, 0.14]$.

A abordagem dinâmica não foi aplicada nos *datasets* originais justamente por conta dessa discrepância de resultados com o APP. Caso o problema seja facilmente resolvido com o *LQ Train*, como a Tabela 7 mostra, uma abordagem de retreino ou até mesmo

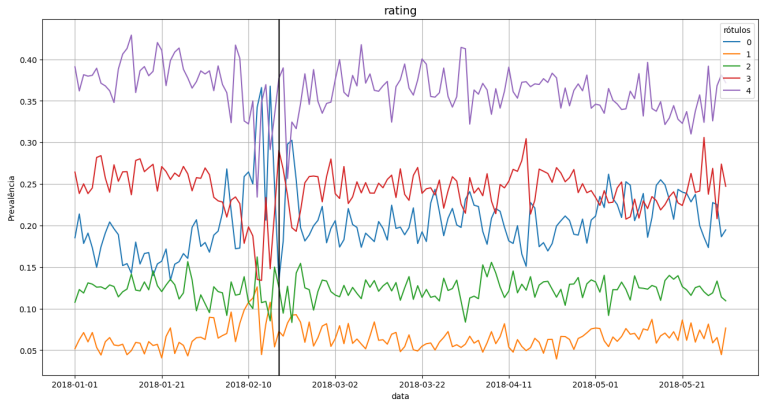


Figura 19 – Prevalência das classes ao longo do tempo do dataset *rating*.

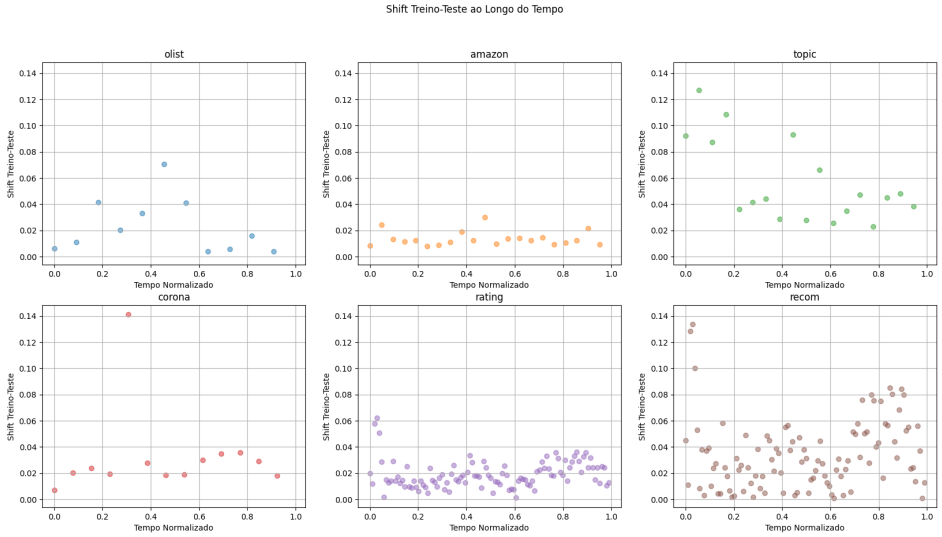


Figura 20 – *Shift* entre treino e teste ao longo dos *batches* de cada conjunto de dados. O eixo x foi normalizado para os *datasets* compartilhassem o mesmo intervalo.

modelos de quantificação não são necessários.

5.2 Datasets Artificiais

Para confrontar os resultados obtidos na Subseção 5.1.2, os *datasets* foram modificados de forma a induzir uma maior variação da prevalência nos conjuntos de teste. A Figura 21 ilustra essa transformação para o *dataset olist*.

Para a transformação, os primeiros *batches* usados como conjunto de treino foram mantidos inalterados, e apenas os conjuntos de teste foram modificados. Dessa forma, ambas as versões de cada *dataset*, original e artificial, possuem a mesma prevalência no conjunto de treino. Além de uma variação maior das prevalências, foi introduzida uma mudança na ordenação das classes, adicionando tendência, positiva e negativa, em determinadas classes. Os conjuntos de teste, ou *batches* de teste, foram modificados utilizando apenas a técnica de *undersampling*, de forma que nenhuma instância teve sua

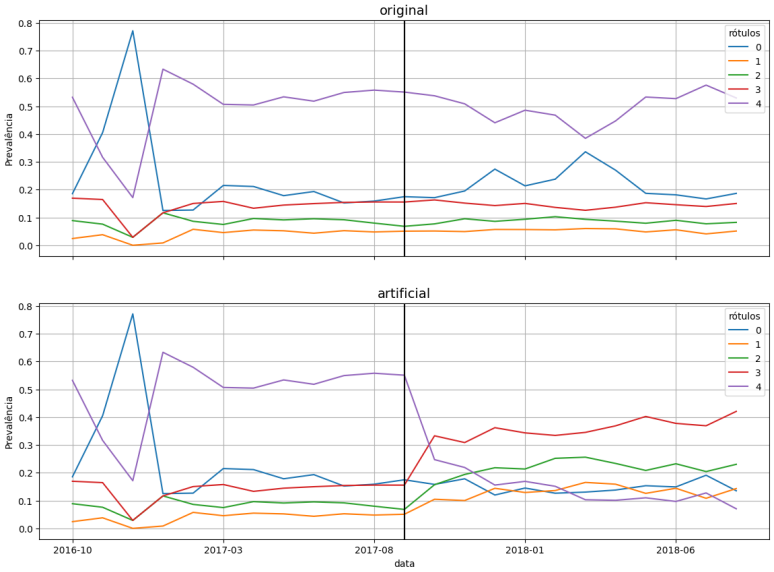


Figura 21 – A figura superior representa o *dataset* original e a figura inferior o *dataset* artificialmente gerado. A reta vertical indica até quando é considerado conjunto de treino.

data original modificada para alterar a prevalência de outro *batch*.

A Figura 22 ilustra o *shift* entre o treino e o teste dos *datasets* artificiais. Diferentemente das versões originais dos conjuntos de dados, os valores de *shift* são numericamente maiores, e estão dispostos no intervalo $[0, 0.5]$.

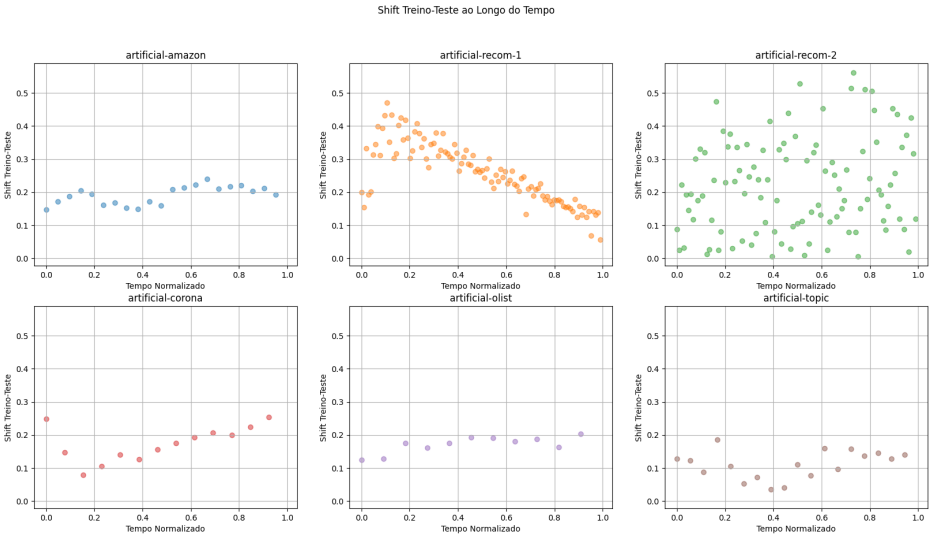


Figura 22 – *Shift* entre treino e teste ao longo dos *batches* de cada *dataset* artificial. O eixo x foi normalizado.

Vale reforçar que essas transformações foram feitas de forma arbitrária, visando introduzir uma variação pressuposta em problemas reais de quantificação, assim como o APP. Porém, diferentemente do APP, os *datasets* artificalmente gerados não apresentam prevalências uniformemente distribuídas.

A Figura 23 ilustra o índice jaccard entre os *batches* de teste e conjunto de treinamento para ambas as versões do *dataset olist*. O índice jaccard é calculado por meio das 200 palavras mais frequentes no *batch*. É possível notar que para a versão artificial o índice jaccard é menor do que na versão original, o que pode indicar uma diferença entre os contextos de treino e de teste.

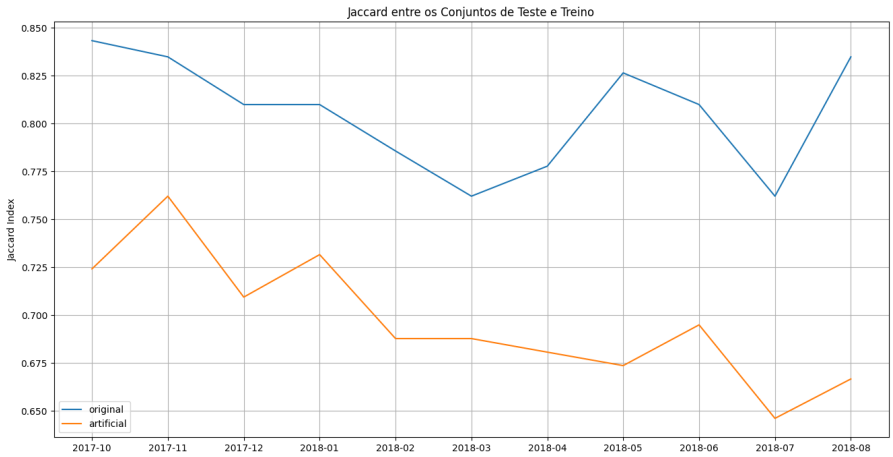


Figura 23 – Índice Jaccard entre os *batches* de teste e treino para o *dataset olist*.

Para os *datasets* artificiais ambas as abordagens, estática e dinâmica, foram avaliadas, e seus resultados estão descritos a seguir.

5.2.1 Estático

A Tabela 9 apresenta o EAM de cada quantificador entre todos os *datasets* artificiais criados.

Tabela 9 – Erro Absoluto Médio de cada quantificador.

Quantificador	EAM
DyS-Topsøe	**0.039148**
SORD	*0.040823
PACC	*0.041702
SMM	*0.043754
ACC	0.048022
CC	0.102001
LQ Balanced	0.103904
PCC	0.109066
LQ Train	0.221458

O quantificador DyS-Topsøe apresentou o melhor EAM, entretanto, os quantificadores SORD, PACC e SMM não apresentaram diferença estatística em relação a ele. A Figura 24 ilustra o ranqueamento médio dos quantificadores.

Comparando a Tabela 9 com a Tabela 7, o comportamento dos quantificadores muda. O PCC, nesse cenário, não apresenta o melhor resultado e aparece somente na penúltima posição - tanto na Tabela 9 quanto na Figura 24. Paralelamente, o *LQ Train* possui a pior média de EAM de todos os quantificadores - o que é esperado devido a própria

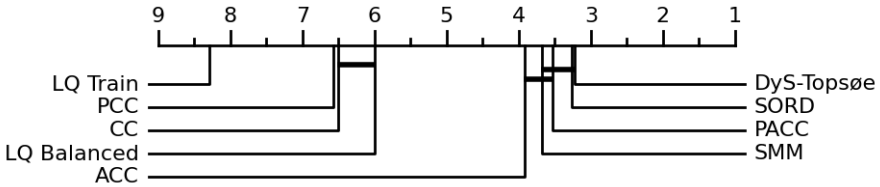


Figura 24 – Ranqueamento médio dos quantificadores. Os modelos ligados por uma linha horizontal não apresentam diferença estatística significativa entre si (teste de *Wilcoxon*).

transformação dos *datasets* artificiais. Além disso, a performance dos quantificadores DyS-Topsøe, SORD e SMM nos *datasets* artificiais se assemelha ao resultado do APP, no sentido de que apresentam os menores erros dentre os quantificadores avaliados. Além disso, o PACC também aparece como um dos modelos estatisticamente semelhante ao DyS-Topsøe.

A Figura 25 apresenta o *boxplot* do EAM de cada quantificador para os *datasets* artificiais. É notável como o intervalo dos valores do *LQ Train* é muito maior do que na Figura 16, assim como o do PCC.

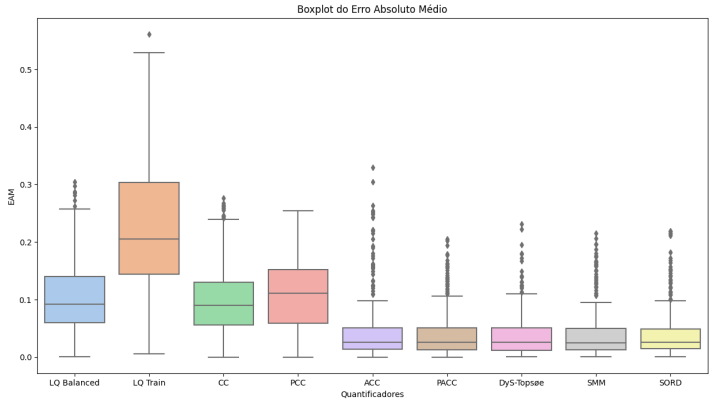


Figura 25 – *Boxplot* do EAM dos quantificadores avaliados.

A Figura 26 apresenta uma comparação entre o EAM e a diferença de distribuição entre o treino e a predição dos *batches* dos *datasets* artificiais. Exceto para os modelos CC e PCC, é menos evidente a correlação entre as duas variáveis como na Figura 18. Diferentemente dos *datasets* originais, os modelos com as melhores performances apresentam predições bem diferentes do treino, *shift alto*, mas com erro baixo, como é o caso do DyS-Topsøe.

A Figura 27 apresenta o comportamento do EAM dos *batches* de teste ao longo do tempo. O eixo x for normalizado para recair sobre o intervalo $[0, 1]$ - onde 0 representa a primeira data e 1 a última data de cada *dataset*.

A performance dos modelos parece ser mais afetada pela diferença de distribuição entre o treino e o teste do que pela passagem do tempo em si. O comportamento dos modelos CC, PCC, *LQ Tain* e *LQ Balanced* acompanha, para o *dataset* artificial *recom-1*

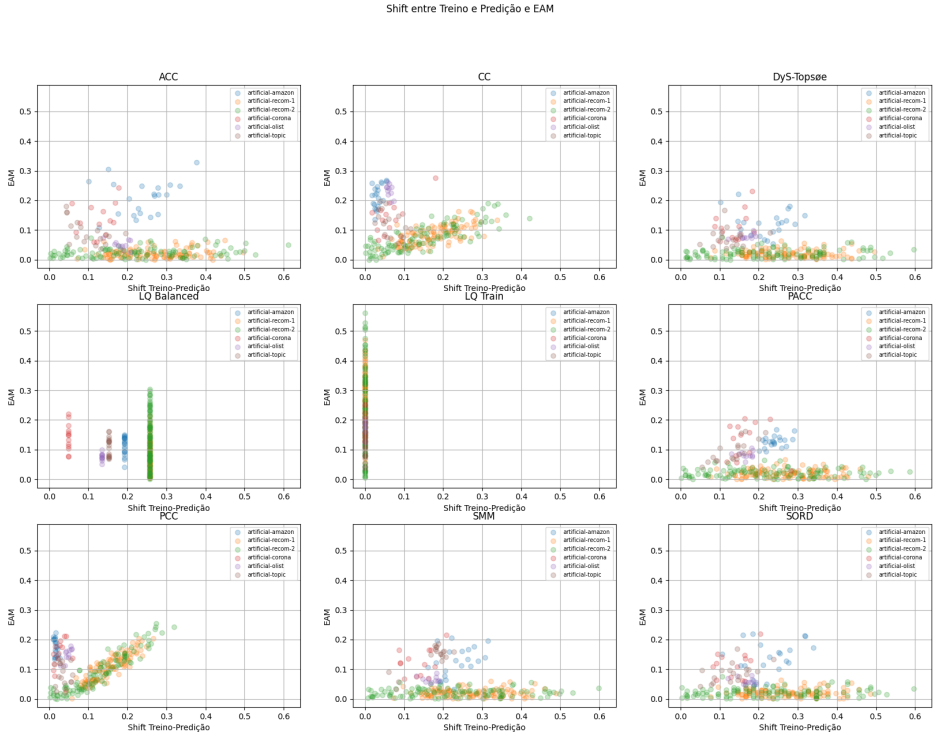


Figura 26 – Comparação do *shift* entre o treino e a predição e o Erro Absoluto Médio da predição.

em amarelo, a diferença de distribuição ilustrada na Figura 30. Essa correlação também aparenta existir para o *dataset* recom-2 (ver Figura 31), onde existe uma maior dispersão dos pontos pelo gráfico.

A Tabela 10 traz a correlação entre a série temporal gerada pela predição dos algoritmos avaliados com a série da prevalência real de cada classe no *dataset artificial-corona*. Juntamente com a Figura 32, é possível perceber que houve casos em que as predições dos quantificadores acompanharam o real comportamento da prevalência, como na classe 4, e casos em que isso não aconteceu, como na classe 1.

Tabela 10 – Correlação das séries temporais preditas com as séries reais de cada classe do *dataset artificial-corona*.

Quantificador	0	1	2	3	4
ACC	-0.040173	-0.156296	0.138192	0.155097	0.812436
CC	-0.132828	0.081866	0.495336	0.230869	0.856735
DyS-Topsøe	0.253016	0.348434	0.736897	0.358238	0.896684
PACC	0.220289	0.610451	0.623597	0.067367	0.887590
PCC	-0.099830	0.762171	0.893498	0.429252	0.875487
SMM	0.419522	0.373396	0.742344	0.211186	0.872142
SORD	0.609571	0.380145	0.711069	0.427166	0.870841

Entretanto, para os conjuntos de dados multi-classe, todos esses valores de correlação, principalmente quebrados por classe, não permitem uma conclusão geral e unificada para o comportamento dos modelos. A única percepção mais clara é de que os modelos com

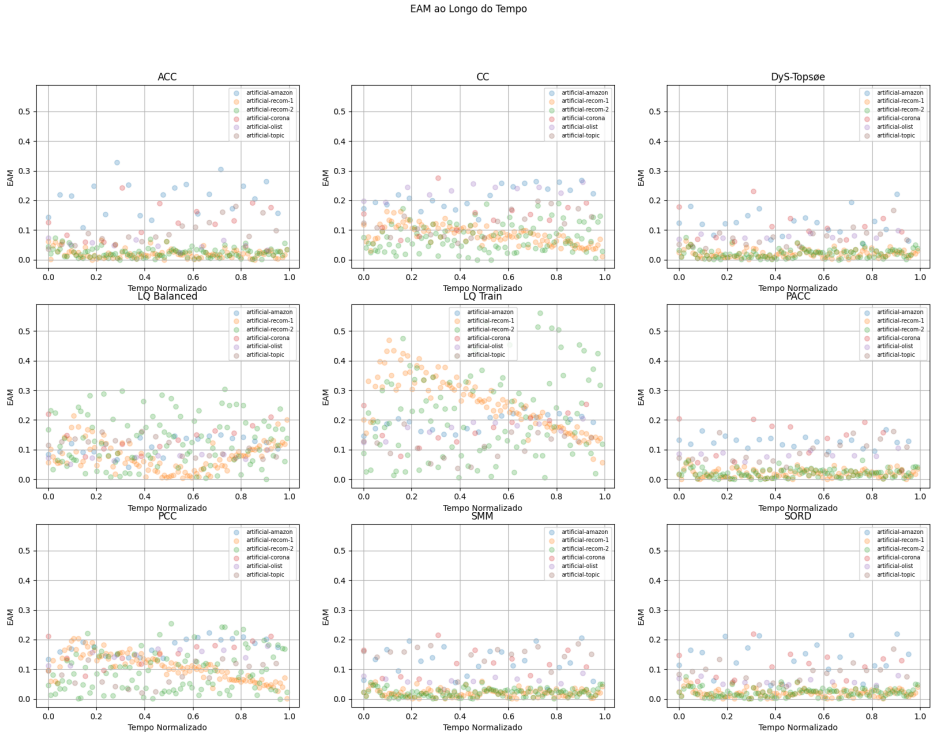


Figura 27 – Comparação entre o Erro Absoluto Médio e a distância temporal em relação ao treino.

os menores erros possuem as maiores correlações, porém essa característica é facilmente inferida através da própria informação de possuir valores de erro baixos.

5.2.2 Dinâmico

Para os *datasets* artificiais, também foi realizado um experimento retreinando de forma histórica os modelos a cada novo *batch* disponível. A Tabela 11 apresenta o EAM de cada quantificador nesta abordagem. Os modelos marcados com um asterisco não apresentam diferença estatística em relação ao melhor modelo, o PACC.

Tabela 11 – Erro Absoluto Médio de cada quantificador.

Quantificador	EAM
PACC	**0.036569**
DyS-Topsøe	*0.037108
SMM	*0.038150
SORD	*0.038294
ACC	0.044895
CC	0.068323
PCC	0.077473
LQ Balanced	0.103904
LQ Train	0.166051

Assim como os outros experimentos, a Figura 28 ilustra o ranqueamento médio dos quantificadores. Os modelos DyS-Topsøe , PACC, SORD e SMM não apresentaram di-

ferença estatística entre si. Vale ressaltar que esse resultado mostra que um modelo de correção, o PACC, possui uma performance tão boa quanto modelos baseados em *matching* de distribuições - considerados estado-da-arte.

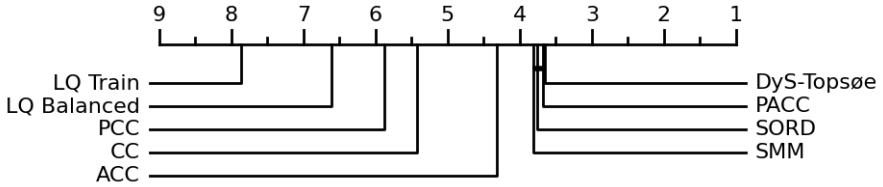


Figura 28 – Ranqueamento médio dos quantificadores. Os modelos ligados por uma linha horizontal não apresentam diferença estatística significativa entre si (teste de *Wilcoxon*).

A Tabela 12 agrega apenas os modelos que obtiveram os melhores resultados em ambas as abordagens, estática e dinâmica, são eles: PACC, DyS-Topsøe, SORD e SMM. Os valores em negrito representam o menor EAM de suas respectivas colunas. Os modelos assinalados com asterisco não apresentaram diferença estatística significativa entre as abordagens estática e dinâmica, apesar do EAM ser numericamente pior nos experimentos com os modelos estáticos.

Tabela 12 – Erro Absoluto Médio dos melhores quantificadores em cada abordagem.

Quantificador	Estático	Dinâmico
PACC	0.041702	**0.036569**
*DyS-Topsøe	**0.039148**	0.037108
*SMM	0.043754	0.038150
*SORD	0.040823	0.038294

Esse resultado sugere que, a partir de determinado tamanho, aumentar o conjunto de treinamento não vai trazer tanto benefício para os quantificadores DyS-Topsøe, SORD e SMM e que é possível obter resultados satisfatórios com quantificadores, e seus respectivos classificadores base, teoricamente desafados ao longo do tempo. Contudo, como a análise da Figura 27 sugere, essa hipótese parece ser válida somente em *datasets* que não possuem mudança de conceito ao longo do tempo.

5.3 *Forecasting*

Nesta seção estão dispostos os resultados para a abordagem descrita na Subseção 4.3.4, que modela o problema de quantificação temporal em um problema de regressão. Na Subseção 5.3.1 são apresentados os resultados do experimento utilizando os *datasets* originais; enquanto que na Subseção 5.3.2 são apresentados os resultados com os *datasets* artificiais.

5.3.1 Datasets Originais

A Tabela 13 apresenta o EAM de cada regressor entre todos os *datasets* avaliados.

Tabela 13 – Erro Absoluto Médio dos regressores nos *datasets* originais.

Regressor	Propagativa	Real
LinearRegression	**0.050617**	**0.031026**
SVR	0.082329	0.045037
MLP	0.129138	0.089855

Assim como a abordagem de quantificação, os regressores apresentam uma performance alta quando avaliados nos *datasets* originais - comparável à peformance dos quantificadores disposta na Tabela 7. Adicionalmente, a técnica de utilizar como *feature* somente os valores reais da série apresenta resultados melhores do que reutilizar a predição dos modelos. Entretanto, vale ressaltar novamente que o comportamento da prevalência não possui uma variação muito grande para os *datasets* originais.

5.3.2 Datasets Artificiais

Da mesma forma que as outras abordagens, os regressores foram avaliados nos *datasets* artificiais. A Tabela 14 consolida o EAM de cada modelo.

Tabela 14 – Erro Absoluto Médio dos regressores nos *datasets* artificiais.

Regressor	Propagativa	Real
MLP	**0.130761**	0.132462
SVR	0.131983	0.143515
LinearRegression	0.174633	**0.104721**

Os modelos apresentam uma performance pior nestes *datasets* do que nos originais, muito provavelmente devido a inserção de variação nos valores das prevalências de cada classe, tornando a tarefa em si mais difícil. Outras características dessa abordagem também impactam nesses resultados, como o tamanho relativamente pequeno da janela de *features* e as séries, para os *datasets* artificiais, não serem estacionárias, ou seja, o teste possui um comportamento diferente do treino.

Outra hipótese para a baixa qualidade dos modelos é a quantidade pequena de instâncias para treinamento juntamente com a reutilização da predição como *feature* de entrada para os regressores. Consequentemente, muitas predições, como é o caso do modelo SVR, acabaram convergindo para um mesmo valor após poucos *batches* de teste, tendo quase nenhuma variação ao longo do tempo.

Neste cenário e nessas configurações, não é interessante utilizar a abordagem de *forecasting*, uma vez que os resultados demonstram uma qualidade inferior a abordagem de quantificação - mesmo quando comparado ao cenário estático.

As tabelas de correlação estão dipostas no Apêndice A.2.

Conclusão

Este trabalho teve por objetivo realizar uma análise sobre o comportamento de modelos de quantificação em conjuntos de dados com uma estrutura temporal definida. Essa análise requer uma configuração experimental específica, que respeite a estrutura mencionada. Além disso, essa mesma estrutura abre espaço para variações de como os modelos podem ser aplicados em problemas dessa natureza.

Diferentemente de muitos trabalhos, a análise conduzida considerou diversos conjuntos de dados multi-classe. Essa característica traz diversos desafios para a avaliação dos modelos, como a quantidade de subconjuntos de teste gerados no protocolo APP e consequentemente o tempo necessário para avaliar os modelos em todos estes subconjuntos. Outro ponto relevante é o impacto que essa quantidade de subconjuntos de teste teve nos testes estatísticos, que sempre rejeitaram a hipótese nula.

Os resultados obtidos demonstram que modelos baseados em correção de classificadores, como o PACC, apresentam bons resultados em *datasets* que não possuem uma variação grande nas prevalências ao longo do tempo. Entretanto, caso o contrário, modelos considerados estado-da-arte, por exemplo o DyS-Topsøe, serão menos suscetíveis à variação e consequentemente melhores.

Outra contribuição interessante deste trabalho foi de que modelos treinados uma única vez são estatisticamente semelhantes às suas versões retreinadas periodicamente. Esse resultado indica que é possível obter bons resultados com menos esforço, principalmente por não necessitar de uma obtenção dos rótulos dos novos dados obtidos ao longo do tempo. Entretanto, não se deve desconsiderar a hipótese de que esse resultado é influenciado pela ausência de uma mudança de conceito nos dados e também que esse resultado foi obtido através dos *datasets* artificialmente gerados.

Adicionalmente, a modelagem do problema para um *forecasting* de séries temporais não se demonstrou efetiva para a maioria dos *datasets* artificiais. Uma hipótese levantada para esse resultado é o tamanho das séries temporais construídas através dos conjuntos de dados, que, apesar de possuírem um grande volume de instâncias de texto, possuem

um comprimento temporal muito reduzido.

Como extensão do trabalho feito nesta pesquisa, destacam-se: (i) investigar *datasets* textuais divididos ao longo do tempo, principalmente a forma como eles são gerados ou coletados; (ii) avaliação de modelos de classificação considerados estado-da-arte no domínio textual, como o BERT; (iii) utilizar, para conjuntos de dados multi-classe, um conjunto de distribuições de classe menor; e (iv) investigar técnicas de *forecasting* mais complexas que consigam auxiliar na quantificação.

Os resultados obtidos através dos conjuntos de dados originais indicam que uma técnica simples como o modelo *LQ Train* consegue obter uma performance melhor que modelos estado-da-arte. Entretanto, essa conclusão parece ser contraditória à própria necessidade de existir a tarefa de quantificação em si. Uma hipótese para essa caracaterística é que a coleta ou distribuição dos textos possa conter um viés indesejado. Uma análise sobre os conjuntos de dados textuais seria interessante para verificar se esse comportamento é inerente aos dados ou não.

Além dos algoritmos de quantificação em si, outra parte fundamental é o classificador base utilizado por várias abordagens. Um dos modelos considerados estado-da-arte, o BERT (DEVLIN et al., 2019) é um forte candidato a ser avaliado em conjuntos de dados textuais. Por obter resultados muito bons em tarefas de classificação, poderia alavancar a performance de quantificadores como o SORD e o DyS-Topsøe.

Os *datasets* multi-classe demandam uma quantidade de distribuições de classe muito grande durante a avaliação de quantificadores no protocolo APP. Uma possível frente de trabalho é investigar métodos que minimizem essa quantidade, mas que não introduzam vieses indesejados. Entretanto, a quantidade de distribuições deve ser minimamente razoável para que seja cumprido o propósito do protocolo, ou seja, consiga-se avaliar os modelos em conjuntos com diferentes prevalências.

Por fim, em relação a tarefa de *forecasting*, há diversas técnicas avançadas disponíveis na literatura tanto para pré-processamento quanto para a predição de séries temporais. É possível explorar tais métodos a fim de encontrar um modelo que consiga ter uma performance tão boa quanto os quantificadores e consiga agregar a informação temporal na predição das prevalências.

Referências

AMATI, G.; BIANCHI, M.; MARCONE, G. Sentiment estimation on twitter. In: **Proceedings of the 5th Italian Information Retrieval Workshop (IIR'14)**. [S.l.: s.n.], 2014. p. 39–50.

BELLA, A. et al. Quantification via probability estimators. In: IEEE. **2010 IEEE International Conference on Data Mining**. [S.l.], 2010. p. 737–742.

BENAVOLI, A.; CORANI, G.; MANGILI, F. Should we really use post-hoc tests based on mean-ranks? **Journal of Machine Learning Research**, v. 17, n. 5, p. 1–10, 2016. Disponível em: <http://jmlr.org/papers/v17/benavoli16a.html>.

CHA, S.-H. Taxonomy of nominal type histogram distance measures. In: **Proceedings of the American Conference on Applied Mathematics**. Stevens Point, Wisconsin, USA: World Scientific and Engineering Academy and Society (WSEAS), 2008. (MATH'08), p. 325–330. ISBN 9789606766473.

CHA, S.-H.; SRIHARI, S. N. On measuring the distance between histograms. **Pattern Recognition**, v. 35, n. 6, p. 1355–1370, 2002. ISSN 0031-3203. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0031320301001182>.

CHAPELLE, O.; CHI, M.; ZIEN, A. A continuation method for semi-supervised svms. In: **Proceedings of the 23rd international conference on Machine learning**. [S.l.: s.n.], 2006. p. 185–192.

CSISZÁR, I.; SHIELDS, P. C. et al. Information theory and statistics: A tutorial. **Foundations and Trends® in Communications and Information Theory**, Now Publishers, Inc., v. 1, n. 4, p. 417–528, 2004.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **J. Mach. Learn. Res.**, JMLR.org, v. 7, p. 1–30, dec 2006. ISSN 1532-4435.

DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186. Disponível em: <https://aclanthology.org/N19-1423>.

ESULI, A.; SEBASTIANI, F. Optimizing text quantifiers for multivariate loss functions. **ACM Trans. Knowl. Discov. Data**, Association for Computing Machinery, New York, NY, USA, v. 9, n. 4, jun 2015. ISSN 1556-4681. Disponível em: <https://doi.org/10.1145/2700406>.

FIRAT, A. Unified framework for quantification. **CoRR**, abs/1606.00868, 2016. Disponível em: <http://arxiv.org/abs/1606.00868>.

FORMAN, G. Counting positives accurately despite inaccurate classification. In: SPRINGER. **European conference on machine learning**. [S.l.], 2005. p. 564–575.

_____. Quantifying trends accurately despite classifier error and class imbalance. In: **Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2006. (KDD '06), p. 157–166. ISBN 1595933395. Disponível em: <https://doi.org/10.1145/1150402.1150423>.

_____. Quantifying counts and costs via classification. **Data Min. Knowl. Discov.**, Kluwer Academic Publishers, USA, v. 17, n. 2, p. 164–206, oct 2008. ISSN 1384-5810. Disponível em: <https://doi.org/10.1007/s10618-008-0097-y>.

FORMAN, G.; KIRSHENBAUM, E.; SUERMONDT, J. Pragmatic text mining: minimizing human effort to quantify many issues in call logs. In: **Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2006. p. 852–861.

GAMA, J. et al. Learning with drift detection. In: BAZZAN, A. L. C.; LABIDI, S. (Ed.). **Advances in Artificial Intelligence – SBIA 2004**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 286–295. ISBN 978-3-540-28645-5.

GAO, W.; SEBASTIANI, F. From classification to quantification in tweet sentiment analysis. **Social Network Analysis and Mining**, v. 6, p. 19:1–19:22, 04 2016.

GIACHANOU, A.; CRESTANI, F. Like it or not: A survey of twitter sentiment analysis methods. **ACM Computing Surveys**, v. 49, p. 1–41, 06 2016.

GONZáLEZ, P. et al. A review on quantification learning. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 50, n. 5, sep 2017. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3117807>.

GONZáLEZ-CASTRO, V.; ALAIZ, R.; ALEGRE, E. Class distribution estimation based on the hellinger distance. **Information Sciences**, v. 218, p. 146–164, 01 2013.

HAN, J.; KAMBER, M.; PEI, J. 1 - introduction. In: HAN, J.; KAMBER, M.; PEI, J. (Ed.). **Data Mining (Third Edition)**. Third edition. Boston: Morgan Kaufmann, 2012, (The Morgan Kaufmann Series in Data Management Systems). p. 1–38. ISBN 978-0-12-381479-1. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780123814791000010>.

HASSAN, W.; MALETZKE, A.; BATISTA, G. Accurately quantifying a billion instances per second. In: **2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)**. [S.l.: s.n.], 2020. p. 1–10.

_____. Pitfalls in quantification assessment. In: **Proceedings of the CIKM 2021 Workshops co-located with 30th ACM International Conference on Information and Knowledge Management (CIKM 2021)**. [S.l.: s.n.], 2021.

ISMAIL-FAWAZ, A. et al. An approach to multiple comparison benchmark evaluations that is stable under manipulation of the comparate set. **arXiv preprint arXiv:2305.11921**, 2023.

JOACHIMS, T. A support vector method for multivariate performance measures. In: **Proceedings of the 22nd International Conference on Machine Learning**. New York, NY, USA: Association for Computing Machinery, 2005. (ICML '05), p. 377–384. ISBN 1595931805. Disponível em: <https://doi.org/10.1145/1102351.1102399>.

KIRITCHENKO, S.; ZHU, X.; MOHAMMAD, S. Sentiment analysis of short informal text. **The Journal of Artificial Intelligence Research (JAIR)**, v. 50, 08 2014.

LI, S. et al. The impact of covid-19 epidemic declaration on psychological consequences: A study on active weibo users. **International Journal of Environmental Research and Public Health**, v. 17, n. 6, 2020. ISSN 1660-4601. Disponível em: <https://www.mdpi.com/1660-4601/17/6/2032>.

LIU, C. et al. Improving sentiment analysis accuracy with emoji embedding. **Journal of Safety Science and Resilience**, v. 2, n. 4, p. 246–252, 2021. ISSN 2666-4496. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666449621000529>.

MALETZKE, A. et al. The importance of the test set size in quantification assessment. In: **Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence**. [S.l.: s.n.], 2021. p. 2640–2646.

_____. Dys: A framework for mixture models in quantification. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2019. v. 33, n. 01, p. 4552–4560.

MALETZKE, A. G.; REIS, D. M. dos; BATISTA, G. E. Quantification in data streams: Initial results. In: IEEE. **2017 Brazilian Conference on Intelligent Systems (BRACIS)**. [S.l.], 2017. p. 43–48.

MILLI, L. et al. Quantification trees. In: IEEE. **2013 IEEE 13th International Conference on Data Mining**. [S.l.], 2013. p. 528–536.

MORAES, S. M. W.; MANSSOUR, I. H.; SILVEIRA, M. S. 7x1-pt: um corpus extraído do twitter para análise de sentimentos em língua portuguesa (7x1-pt: a corpus extracted from twitter for sentiment analysis in portuguese language). In: **STIL**. [S.l.: s.n.], 2015.

MOREO, A.; SEBASTIANI, F. Tweet sentiment quantification: An experimental re-evaluation. **PLoS One**, Public Library of Science San Francisco, CA USA, v. 17, n. 9, p. e0263449, 2022.

NAKOV, P. et al. Semeval-2016 task 4: Sentiment analysis in twitter. **CoRR**, abs/1912.01973, 2019. Disponível em: <http://arxiv.org/abs/1912.01973>.

NICULESCU-MIZIL, A.; CARUANA, R. Predicting good probabilities with supervised learning. In: **Proceedings of the 22nd International Conference on Machine Learning**. New York, NY, USA: Association for Computing Machinery, 2005. (ICML '05), p. 625–632. ISBN 1595931805. Disponível em: <https://doi.org/10.1145/1102351.1102430>.

OWOPUTI, O. et al. Improved part-of-speech tagging for online conversational text with word clusters. In: **Proceedings of the 2013 conference of the North American chapter of the association for computational linguistics: human language technologies**. [S.l.: s.n.], 2013. p. 380–390.

REIS, D. M. dos et al. Fast unsupervised online drift detection using incremental kolmogorov-smirnov test. In: **Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 1545–1554. ISBN 9781450342322. Disponível em: <https://doi.org/10.1145/2939672.2939836>.

RUBNER, Y.; TOMASI, C.; GUIBAS, L. J. The earth mover's distance as a metric for image retrieval. **Int. J. Comput. Vision**, Kluwer Academic Publishers, USA, v. 40, n. 2, p. 99–121, nov 2000. ISSN 0920-5691. Disponível em: <https://doi.org/10.1023/A:1026543900054>.

SAIF, H.; HE, Y.; ALANI, H. Semantic sentiment analysis of twitter. In: SPRINGER. **The Semantic Web–ISWC 2012: 11th International Semantic Web Conference, Boston, MA, USA, November 11-15, 2012, Proceedings, Part I 11**. [S.l.], 2012. p. 508–524.

SCHUMACHER, T.; STROHMAIER, M.; LEMMERICH, F. A comparative evaluation of quantification methods. **CoRR**, abs/2103.03223, 2021. Disponível em: <https://arxiv.org/abs/2103.03223>.

SEBASTIANI, F. Evaluation measures for quantification: An axiomatic approach. **Information Retrieval Journal**, Springer, v. 23, n. 3, p. 255–288, 2020.

Apêndices

APÊNDICE A

Material Complementar

A.1 *Datasets* Artificiais

Nesta seção estão dispostos as transformações feitas nos *datasets* originais para obtermos uma maior variação de prevalência ao longo do tempo. As Figuras 29, 30, 31 ,32, 33 e 34 apresentam a prevalência original de cada *dataset* na figura superior e a prevalência gerada artificialmente na figura inferior. A reta vertical indica até quando é considerado conjunto de treino.

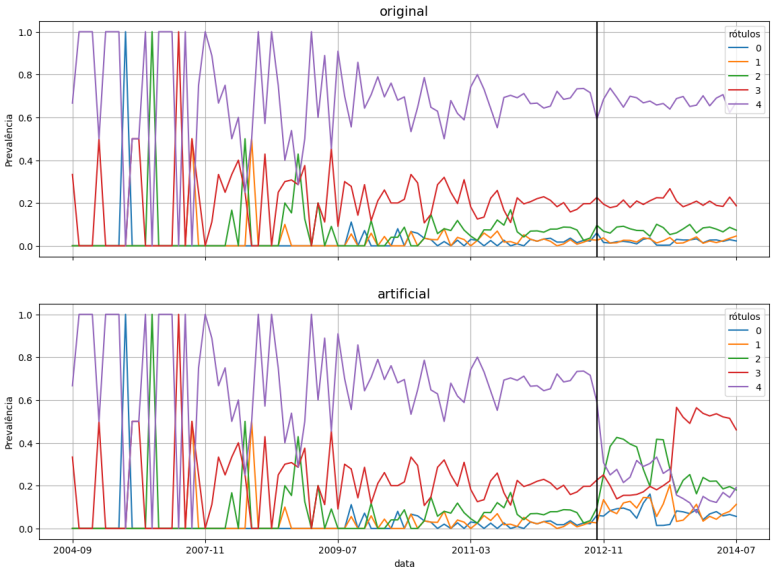


Figura 29 – Transformação do *dataset* amazon.

Figura 30 – Transformação do *dataset* b2w (versão 1).

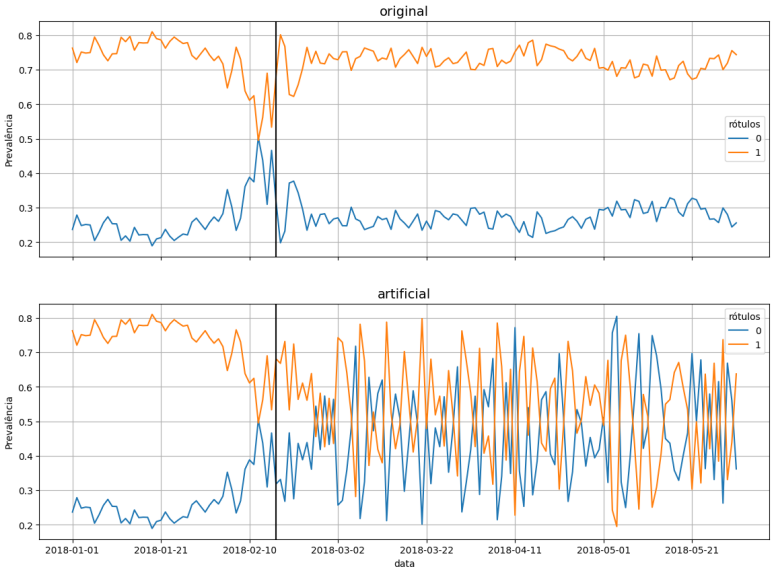


Figura 31 – Transformação do *dataset* b2w (versão 2).

A.2 Correlação das Predições dos Quantificadores nos *Datasets* Artificiais

Nesta seção estão dispostas as correlações entre as predições e as prevalências reais de cada classe para os conjuntos de dados utilizados neste trabalho. As Tabelas 15, 16, 17, 18, 19 e 20 apresentam os valores das correlações com os quantificadores dispostos nas linhas e os rótulos ou classes pertecentes ao *dataset* na colunas.

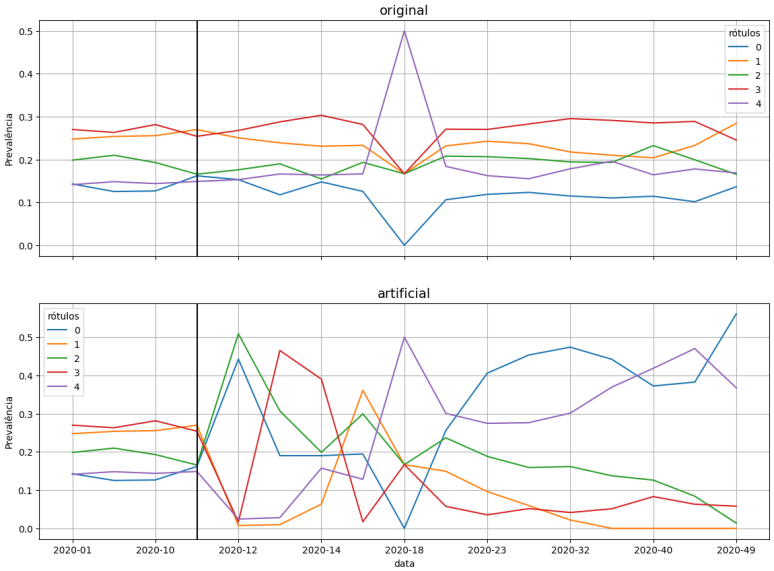


Figura 32 – Transformação do *dataset* corona.

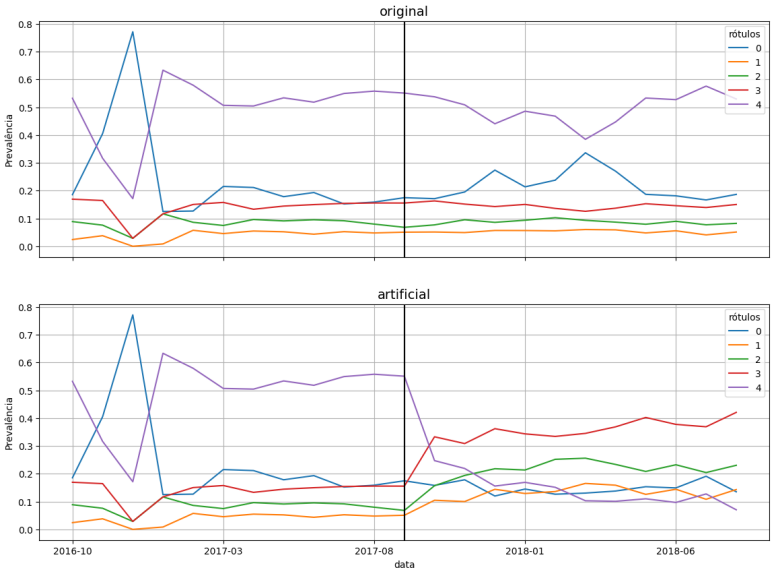


Figura 33 – Transformação do *dataset* olist.

Tabela 15 – Correlação dos quantificadores no *dataset olist*.

Quantificador	0	1	2	3	4
ACC	-0.040173	-0.156296	0.138192	0.155097	0.812436
CC	-0.132828	0.081866	0.495336	0.230869	0.856735
DyS-Topsøe	0.253016	0.348434	0.736897	0.358238	0.896684
PACC	0.220289	0.610451	0.623597	0.067367	0.887590
PCC	-0.099830	0.762171	0.893498	0.429252	0.875487
SMM	0.419522	0.373396	0.742344	0.211186	0.872142
SORD	0.609571	0.380145	0.711069	0.427166	0.870841

Figura 34 – Transformação do *dataset* topic.

Tabela 16 – Correlação dos quantificadores no *dataset amazon*.

Quantificador	0	1	2	3	4
ACC	0.372743	0.021766	0.275320	-0.093126	-0.440143
CC	0.308844	0.289934	0.294449	-0.227574	-0.304631
DyS-Topsøe	0.581101	0.367769	0.374327	0.274285	-0.259792
PACC	0.588254	0.522447	0.210548	0.360971	-0.476777
PCC	0.491943	0.710575	0.289124	-0.041732	-0.404484
SMM	0.614493	0.405218	0.305215	0.094016	-0.363983
SORD	0.592205	0.355320	0.290012	0.091537	-0.394731

Tabela 17 – Correlação dos quantificadores no *dataset recom-1*.

Quantificador	0	1
ACC	0.965352	0.965352
CC	0.965352	0.965352
DyS-Topsøe	0.973848	0.973848
PACC	0.971591	0.971591
PCC	0.972358	0.972358
SMM	0.972866	0.972866
SORD	0.972640	0.972640

Tabela 18 – Correlação dos quantificadores no *dataset recom-2*.

Quantificador	0	1
ACC	0.983081	0.983081
CC	0.983081	0.983081
DyS-Topsøe	0.987798	0.987798
PACC	0.987825	0.987825
PCC	0.987520	0.987520
SMM	0.987899	0.987899
SORD	0.987458	0.987458

Tabela 19 – Correlação dos quantificadores no *dataset corona*.

Quantificador	0	1	2	3	4
ACC	0.488193	0.492574	0.833653	0.409086	0.843852
CC	0.735991	0.187135	0.804328	0.260775	0.784579
DyS-Topsøe	0.857723	0.362703	0.815052	0.609403	0.840814
PACC	0.778988	0.409784	0.779386	0.465479	0.884243
PCC	0.833125	0.349229	0.793557	0.381968	0.927330
SMM	0.851416	0.385470	0.796497	0.561973	0.890423
SORD	0.860767	0.374168	0.822393	0.622702	0.887477

Tabela 20 – Correlação dos quantificadores no *dataset topic*.

Quantificador	0	1	2	3	4	5
ACC	Indefinido	0.133434	0.681584	0.870301	0.892555	0.293074
CC	Indefinido	0.131125	0.745230	0.875111	0.909180	0.093880
DyS-Topsøe	0.495243	0.460006	0.856130	0.868537	0.954365	0.409841
PACC	0.477305	0.243276	0.820268	0.817440	0.932195	0.094062
PCC	0.404613	0.515953	0.766678	0.899273	0.936119	0.447827
SMM	0.119398	0.451132	0.680894	0.860770	0.790318	0.372529
SORD	0.056913	0.396711	0.784687	0.828089	0.933009	0.073641