UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Thalita Firmo Drumond

Coclusterização em comitês de filtros colaborativos e em análise de atividade cerebral

Campinas

2016

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Thalita Firmo Drumond

Coclusterização em comitês de filtros colaborativos e em análise de atividade cerebral

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestra em Engenharia Elétrica, na Área de Engenharia de Computação.

Orientador: Prof. Dr. Fernando José Von Zuben

Este exemplar corresponde à versão final da dissertação defendida pela aluna Thalita Firmo Drumond, e orientada pelo Prof. Dr. Fernando José Von Zuben

_____

Campinas
2016

Agência(s) de fomento e nº(s) de processo(s): FAPESP, 2014/11125-1

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

D845c

Drumond, Thalita Firmo, 1989-
Coclusterização em comitês de filtros colaborativos e em análise de atividade cerebral / Thalita Firmo Drumond. – Campinas, SP : [s.n.], 2016.

Orientador: Fernando José Von Zuben.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Aprendizado de máquina. 2. Mineração de dados (Computação). 3. Reconhecimento de padrões. 4. Sistemas de recomendação (Filtragem da informação). 5. Mapeamento cerebral. I. Von Zuben, Fernando José,1968-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Co-clustering in collaborative filtering committees and in brain activity analysis
Palavras-chave em inglês:
Machine learning
Data Mining (computing)
Pattern recognition
Recommender systems (Information filtering)
Brain mapping
Área de concentração: Engenharia de Computação
Titulação: Mestra em Engenharia Elétrica
Banca examinadora:
Fernando José Von Zuben [Orientador]
Guilherme Palermo Coelho
Romis Ribeiro de Faissol Attux
Data de defesa: 29-08-2016
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA – DISSERTAÇÃO DE MESTRADO

Candidata: Thalita Firmo Drumond RA 082893
Data de defesa: 29 de agosto de 2016

Título da dissertação: "Coclusterização em comitês de filtros colaborativos e em análise de atividade cerebral"

Prof. Dr. Fernando José Von Zuben (Presidente, FEEC/UNICAMP)
Prof. Dr. Guilherme Palermo Coelho (FT/ Unicamp)
Prof. Dr. Romis Ribeiro de Faissol Attux (FEEC/Unicamp)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no processo de vida acadêmica da aluna.

Dedico este trabalho à minha família.

Agradecimentos

Agradeço à FAPESP (processo nº 2014/11125-1) pelo apoio financeiro para a realização deste mestrado.

Agradecimentos pessoais

Muita gente, de uma forma ou de outra, me ajudou a chegar até aqui. Família, amigos, professores, mais gente do que eu posso me lembrar de citar aqui de maneira justa.

Agradeço a meus pais Yara e Ton, minha irmãzinha Isabela (você cresceu mas será sempre minha maninha ^^), que sempre me deram apoio nos meus estudos, na decisão de vir para a Unicamp lá em 2008, na escolha pela carreira acadêmica, mesmo com esses mais de 2 mil quilômetros de distância.

Agradeço a meu namorado Cássio, meu parceiro no trabalho e na vida. Obrigada pela companhia nos fins de semana no laboratório, pelas madrugadas trabalhando, pelo apoio nos momentos de dificuldade que sempre vêm para quem está fazendo pós-graduação. E tantas coisas mais que não cabe aqui descrever. A vida com certeza é muito melhor com você.

Agradeço ao prof. Fernando pela oportunidade de realizar esse mestrado. Suas aulas, seu apoio e sua orientação sem dúvida muito contribuíram para meu crescimento acadêmico e meu futuro profissional.

Agradeço aos colegas do LBiC (e adjacências =)), que me acolheram abertamente nesses anos de mestrado. Foram muitos cafés, GRUDEs, almoços e outros momentos de confraternização, além de discussões sobre nossos trabalhos, dicas e colaborações que sempre me ajudaram muito.

Agradeço aos colegas da graduação (aqui e na França) que me acompanharam até nos fins de semana no bitolódromo/SIFEEC/870/ROSE/..., terminando projetos e estudando para aquelas provas mais temidas. Chegar à formatura não foi fácil, mas ajudou muito poder contar com vocês.

Com certeza tem mais gente para agradecer, mas se fosse para ser realmente justa, acho que ia citar até a Tia Zizi da alfabetização, e mesmo assim não teria acabado. A todos que interferiram positivamente na minha vida e na minha carreira: Muito obrigada!

Resumo

Os avanços em tecnologia de informação têm promovido um crescimento acentuado em geração e armazenamento de dados, produzidos praticamente em todos os campos de atividade humana. Com isso têm surgido demandas específicas de análise e extração de conhecimento dessa massa de dados, buscando um melhor entendimento dos processos envolvidos, a detecção de tendências e o suporte a decisões nas mais variadas áreas. Nesse contexto, buscam-se o estudo e o desenvolvimento de técnicas de aprendizado de máquina e ferramentas de mineração de dados eficientes e escaláveis. Uma dessas ferramentas é a coclusterização, uma técnica que visa agrupar simultaneamente objetos e seus atributos, encontrando grupos que apresentem algum padrão de coerência interna. Essa família de algoritmos tem sido aplicada em dados de expressão gênica, a fim de identificar grupos de genes com padrões de expressão coerentes sob um conjunto de condições. Existem também vários trabalhos usando a coclusterização para filtragem colaborativa, uma abordagem comumente usada para sistemas de recomendação, visando sugerir conteúdos, como livros e filmes, que possam ser de interesse para um usuário. Neste trabalho, diferentes técnicas de coclusterização foram exploradas em dois domínios de aplicação. Primeiramente, um *framework* de filtragem colaborativa, que utiliza uma fatoração booleana de matrizes baseada em coclusterização, foi estendido através de uma abordagem de *ensemble* robusta e escalável, utilizando projeções aleatórias para a redução de dimensão e um método de vizinhança aproximada. A agregação de cada uma das técnicas foi estudada separadamente, com experimentos em bases de dados reais usualmente adotadas na literatura. O *ensemble* proposto também foi comparado com técnicas tradicionais e com o estado-da-arte, apresentando resultados competitivos. Em uma segunda linha de trabalho, uma técnica de coclusterização contígua foi aplicada em séries temporais de atividade cerebral, buscando encontrar padrões de atividade temporal coerente entre regiões do cérebro. Os padrões encontrados foram utilizados para construir um mapa cerebral funcional dinâmico, expresso por padrões de conectividade que evoluem com o tempo. Os mapas funcionais obtidos são relevantes para a visualização dos numerosos padrões encontrados pelo algoritmo de coclusterização, permitindo a discriminação entre pacientes e controles.

Palavras-chaves: aprendizado de máquina; coclusterização; fatoração booleana de matrizes; filtragem colaborativa; coclusterização contígua; análise de atividade cerebral.

Abstract

The advances in information technology are promoting the generation and storage of an ever-increasing amount of data, produced by virtually all fields of human activity. So, there are distinguished demands for the analysis and knowledge extraction of these data, aiming at better understanding the involved processes, detecting trends and supporting decisions in the most varied fields. In this context, the study and development of scalable and efficient data mining tools and machine learning frameworks are highly desirable. One of such techniques is co-clustering, which aims to cluster simultaneously objects and their attributes, locating groups presenting some internal coherence pattern. This family of algorithms has been commonly applied to gene expression data, in order to identify groups of genes with coherent expression patterns under a group of conditions. There are also several works using co-clustering for collaborative filtering, an approach commonly used for recommender systems, which aims to suggest objects or contents, such as movies or books, that might interest a user. In this work, different co-clustering techniques were explored in two different application domains. First, a collaborative filtering framework, using a co-clustering-based matrix factorization technique, was extended through a robust and scalable ensemble approach using random projections for dimensionality reduction and approximate nearest neighbors. The agregation of each technique was studied individually, with experiments on real-world datasets commonly used in the litterature. The proposed ensemble was also compared to traditional and state-of-the-art techniques, exhibiting competitive results. Secondly, a contiguous co-clustering technique was applied to different neuronal brain activity time-series, looking for coherent temporal activity patterns between brain regions. Patterns found were used to construct a dynamic brain mapping, expressed by connectivity patterns that evolve with time. Those obtained functional maps are relevant for visualization of the numerous patterns found by the co-clustering algorithm, allowing the discrimination between patients and controls.

Keywords: machine learning; co-clustering; Boolean matrix factorization; collaborative filtering; contiguous co-clustering; brain activity analysis.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Com a crescente disponibilidade de recursos computacionais, a geração, o processamento e a armazenagem de dados têm crescido de forma acentuada e continuada. Com este crescimento, aumenta a demanda por soluções computacionais capazes de analisar e extrair conhecimento útil desses dados. Para fazer face a essa demanda, a pesquisa e o desenvolvimento de métodos de aprendizado de máquina e mineração de dados têm acompanhado esse crescimento, buscando ferramentas com melhor desempenho, escaláveis e adaptadas aos mais diferentes domínios de aplicação.

A coclusterização é uma das várias técnicas empregadas para mineração de padrões em bases de dados. Um exemplo muito comum são as matrizes de dados, cujas linhas correspondem a objetos e cujas colunas correspondem aos atributos desses objetos. Cada elemento da matriz, portanto, vai indicar o valor que um certo atributo assume para um certo objeto. Diferentemente da clusterização, que busca agrupar um subconjunto de objetos dessa matriz de dados que apresentem uma coerência ao longo de todos os seus atributos, a coclusterização visa agrupar um subconjunto de objetos que tenham coerência em um subconjunto de atributos. Portanto, a coclusterização busca encontrar simultaneamente subgrupos de linhas e colunas que apresentem algum padrão de coerência.

Essa família de algoritmos tem sido aplicada principalmente em dados de expressão gênica, geralmente expressos por matrizes de genes $\times$ condições. Busca-se encontrar grupos de genes com padrões de expressão coerentes ao longo de várias condições (MADEIRA; OLIVEIRA, 2004). Também existem aplicações analisando a dinâmica da expressão de genes, em matrizes cujas linhas são séries temporais de expressão gênica. Nesse cenário, a coclusterização busca grupos de genes com uma dinâmica de expressão similar em alguma janela contígua de tempo. Essa técnica em especial — a coclusterização contígua — pode ser aplicada em qualquer contexto de busca de padrões em séries temporais simultâneas, e foi sugerido que seu uso pudesse ser relevante para outros tipos de dados biológicos (MADEIRA *et al.*, 2010).

Existem também diversos trabalhos utilizando coclusterização e técnicas correlatas para a filtragem colaborativa, uma das abordagens mais utilizadas na realização de sistemas de recomendação. Considerando um cenário em que existem clientes interessados em serviços ou produtos, que aqui serão denominados itens, um sistema de recomendação envolve a sugestão de itens que possam atender maximamente aos interesses dos clientes. O grande desafio é realizar esta recomendação personalizada a partir apenas de dados disponíveis. Com isso, a relevância de cada item vai depender, por exemplo, do histórico de compras de um cliente numa loja virtual, de avaliações de itens, de definição de preferências e restrições e de classificação de padrões (RICCI *et al.*, 2011).

Diversas abordagens têm sido utilizadas na solução desse problema. Sistemas baseados em modelos de fatores latentes, inferidos com técnicas de fatoração de matrizes, se tornaram o estado-da-arte (JANNACH *et al.*, 2010; KOREN; BELL, 2011; SYMEONIDIS; PANAGIOTIS, 2016). Tais métodos têm tido sucesso no cenário de filtragem colaborativa. Mais recentemente, modelos de fatoração booleana de matrizes baseada em coclusterização produziram resultados promissores (IGNATOV *et al.*, 2014). Esse método pode ser particularmente interessante para dados binários, pois as matrizes de fatores e a reconstrução são binárias, o que não acontece com fatorações baseadas em SVD ou fatorações não-negativas.

Neste trabalho, diferentes técnicas de coclusterização foram aplicadas em dois domínios distintos. Primeiramente, foi proposto uma *framework* de ensemble de projeções aleatórias, como extensão do trabalho de Ignatov *et al.* (2014), visando maior robustez das recomendações e mantendo a escalabilidade. A projeção de dados em um subespaço selecionado aleatoriamente é uma técnica de redução de dimensionalidade eficiente, que preserva aproximadamente as distâncias par-a-par. Entretanto, o uso de projeções aleatórias pode levar a resultados muito diferentes (FERN *et al.*, 2006), motivando a criação de um *ensemble*. Além disso, age como uma forma de gerar múltiplas perspectivas dos dados, o que tende a trazer uma diversidade importante para os componentes do ensemble. Também foi estudado o uso de um método de vizinhança aproximada na escalabilidade e acurácia do sistema de recomendação (BAWA *et al.*, 2005). Foram realizados testes na base Movielens 100k[1] e também na base Delicious 2k, comumente usadas na literatura.

A segunda linha de trabalho trata de análise de atividade cerebral, e o vincula ao projeto CEPID (Centros de Pesquisa, Inovação e Difusão) intitulado BRAINN (Instituto Brasileiro de Neurociência e Neurotecnologia) [2]. As questões centrais deste projeto CEPID referem-se à investigação biológica dos mecanismos básicos que levam à epilepsia e ao derrame cerebral, assim como dos mecanismos de lesão e progressão associados a essas doenças, e estão relacionadas à prevenção, ao tratamento e à reabilitação.

Nesse contexto, foi proposta a aplicação da coclusterização contígua como método de identificação de padrões em séries temporais de atividade cerebral. Nesse contexto, a coclusterização contígua permite a identificação de regiões co-funcionais, ou seja, regiões apresentando padrões de atividade temporal coerente. Em seguida, foi proposto o uso da informação obtida pela coclusterização — os coclusters agrupando regiões em janelas de tempo — na geração de um mapeamento funcional dinâmico do cérebro. Foram gerados modelos em rede, nos quais os nós são regiões do cérebro e as arestas ligam regiões de um mesmo cocluster na janela de tempo indicada por ele. Esse mapeamento é uma maneira relevante e intuitiva de visualizar a informação contida nos padrões espaço-temporais encontrados pela coclusterização. Esta técnica de mapeamento funcional também pode ser utilizado para a análise da atividade

[1] <http://grouplens.org/datasets/movielens/>

[2] Processo Fapesp no. 2013/07559-3: <http://cepid.fapesp.br/pdf/BRAINN_11.pdf>

cerebral sob diferentes circunstâncias. Foram analisadas métricas topológicas dessas redes e foi verificada a sua capacidade de discriminar entre indivíduos do grupo de controle e do grupo de pacientes.

1.1 Organização da dissertação

Esta dissertação se divide em duas partes. Na primeira, serão apresentados os fundamentos e conceitos utilizados no desenvolvimento do trabalho. São os seguintes:

Capítulo 2 Apresenta a técnica de coclusterização, juntamente com os algoritmos relacionados que foram utilizados nesse trabalho.

Capítulo 3 Apresenta uma revisão da área de sistemas de recomendação, com foco em filtragem colaborativa, de maior interesse para esse trabalho.

Capítulo 4 Apresenta a teoria de busca aproximada de vizinhos com o uso de funções de *hashing* sensíveis à localidade (em inglês, *locally sensitive hashing* - LSH), juntamente com alguns algoritmos.

Capítulo 5 Discorre sobre o uso de projeções aleatórias para a redução de dimensão em bases de dados.

Capítulo 6 Apresenta uma visão geral da análise de atividade cerebral dentro do paradigma de redes complexas.

Na segunda parte, são apresentadas as duas propostas de trabalho realizadas durante esse mestrado.

Capítulo 7 Detalha a metodologia proposta no contexto de sistemas de recomendação, bem como os experimentos realizados e os resultados obtidos.

Capítulo 8 Apresenta a proposta de análise de atividade cerebral via coclusterização contígua, os experimentos realizados e os resultados obtidos.

Por fim, no capítulo 9 são apresentadas considerações finais gerais e possibilidades futuras para as propostas aqui formalizadas, seguidas pelas referências bibliográficas citadas e/ou consultadas no desenvolvimento deste trabalho.

Parte I

Fundamentos

2 Coclusterização

A clusterização é uma técnica de aprendizado de máquina cujo objetivo é descobrir padrões coerentes em dados não rotulados. Baseando-se nesses padrões exibidos pelos objetos — como proximidade ou similaridade — a clusterização é usada para separá-los em diferentes grupos. Por exemplo, no algoritmo *k-means* (MACQUEEN, 1967), um dos mais utilizados, o critério de separação em k grupos é minimizar a soma das distâncias entre o ponto médio (centroide) e os objetos pertencentes ao respectivo grupo. Na clusterização, objetos em um mesmo grupo tendem a apresentar coerência em todos os seus atributos.

Na coclusterização, o agrupamento é executado não somente sobre os objetos mas também sobre seus atributos, de forma simultânea (MADEIRA; OLIVEIRA, 2004). Dessa forma, um cocluster é composto de um subconjunto de objetos e de um subconjunto de atributos nos quais esses objetos apresentaram alguma relação. Considerando uma matriz de dados, onde linhas são objetos e colunas são atributos, a coclusterização — nesse caso também chamada de biclusterização (CHENG; CHURCH, 2000) — encontra subconjuntos de linhas e colunas que determinam uma sub-matriz cujas entradas exibem algum padrão de coerência interna.

Note que o termo biclusterização faz sentido quando temos apenas duas dimensões (objetos e atributos). Porém, caso haja outras dimensões expressando o relacionamento entre os objetos considerados, o termo coclusterização — mais genérico — é o mais adequado. Neste trabalho, trataremos apenas de dados diádicos, que podem ser representados por uma matriz, de modo que coclusters e biclusters são equivalentes.

A possibilidade de agrupar dados que são relacionados apenas em um subconjunto de atributos torna a coclusterização mais versátil que a clusterização tradicional, que exige coerência em todos os atributos. Mesmo que fosse aplicado algum pré-processamento para seleção de atributos, a coclusterização tem a vantagem de automaticamente selecionar os atributos relevantes para cada grupo, que não precisam ser os mesmos de um grupo a outro. Além disso, um mesmo objeto pode compor múltiplos coclusters, os quais são constituídos por subconjuntos distintos de atributos.

Para a definição formal de um cocluster, vamos tomar uma matriz $M \in \mathbb{R}^{n \times m}$, que pode ser definida por seu conjunto de linhas $X = \{x_1, x_2, \ldots, x_n\}$ e seu conjunto de colunas $Y = \{y_1, y_2, \ldots, y_m\}$. Considere $I \subseteq X$ e $J \subseteq Y$ como sendo subconjuntos das linhas e colunas da matriz M, respectivamente. O cocluster $M_{IJ} = (I, J)$ representa uma submatriz de M, contendo apenas os elementos m_{ij} pertencentes ao subconjunto de linhas I e ao subconjunto de colunas J de A. Esses elementos m_{ij}, tais que $i \in I$ e $j \in J$, compõem um cocluster por apresentarem algum padrão de coerência interna. Se a inclusão de qualquer linha $i \in X$ ou coluna $j \in Y$ violar o padrão de coerência de um cocluster $M_{IJ} = (I, J)$, esse cocluster é dito maximal.

A enumeração de todos os coclusters maximais em uma base de dados é um problema NP-completo. Por isso, comumente são utilizados algoritmos heurísticos, apesar de existirem algumas propostas enumerativas computacionalmente eficientes (VERONEZE *et al.*, 2016).

2.1 Tipos de coclusters

2.1.1 Tipo de coerência

Existem quatro principais tipos de coerência interna que um cocluster pode apresentar (MADEIRA; OLIVEIRA, 2004; SABER; ELLOUMI, 2015):

Valores constantes Nesse caso, todos os seus elementos m_{ij} apresentam o mesmo valor μ — um cocluster de valores constantes perfeito. Pode ser considerado também o caso com ruído, onde todos os elementos têm o mesmo valor mais um ruído ω_{ij}. De maneira geral, é definido pela equação $m_{ij} = \mu + \omega_{ij}$.

Valores constantes nas linhas ou nas colunas Nesse modelo, dado que uma das linhas possui um valor constante μ, as outras linhas podem ser obtidas com a adição ou multiplicação de um termo α_i mais um ruído ω_{ij}. Dessa forma, os elementos podem ser obtidos por $m_{ij} = \mu + \alpha_i + \omega_{ij}$ ou $a_{ij} = \mu \times \alpha_i \times \omega_{ij}$. O mesmo raciocínio se aplica ao cocluster com valor constante nas colunas. Para este caso, as equações são $a_{ij} = \mu + \beta_j + \omega_{ij}$, ou $a_{ij} = \mu \times \beta_j \times \omega_{ij}$.

Valores coerentes Nesse tipo de cocluster, as variações entre um elemento e o seguinte são iguais nas linhas e/ou colunas, mais um ruído ω_{ij}. Essa variação pode ser aditiva ou multiplicativa. No modelo aditivo, os elementos são determinados pela equação $m_{ij} = \mu + \alpha_i + \beta_j + \omega_{ij}$. Já no modelo multiplicativo, o cocluster tem seus elementos obtidos pela equação $m_{ij} = \mu \times \alpha_i \times \beta_j \times \omega_{ij}$. Em ambos os casos, α_i é o fator de variação comum aos elementos da linha i, e β_j o fator comum aos elementos da coluna j. Note que esse modelo é uma generalização dos anteriores. Se $\alpha_i = 0$ (ou 1 para o caso multiplicativo), tem-se um cocluster com valores constantes nas colunas. Analogamente, Se $\beta_j = 0$ (ou 1 para o caso multiplicativo), tem-se um cocluster com valores constantes nas linhas. Se ambos são nulos, recai-se no caso do cocluster com valores constantes.

Evolução coerente Nesse tipo de cocluster, busca-se um subconjunto de linhas e/ou colunas que apresentem uma evolução coerente, ou seja, cujos valores aumentem e decresçam em colunas/linhas correspondentes. Diferentemente do cocluster de valores coerentes, não é necessário que as variações sejam constantes.

A Figura 1 ilustra esses tipos de coerência interna em coclusters.

1,0	1,0	1,0	1,0
1,0	1,0	1,0	1,0
1,0	1,0	1,0	1,0
1,0	1,0	1,0	1,0

(a)

1,0	2,0	3,0	4,0
1,0	2,0	3,0	4,0
1,0	2,0	3,0	4,0
1,0	2,0	3,0	4,0

(b)

1,0	1,0	1,0	1,0
2,0	2,0	2,0	2,0
3,0	3,0	3,0	3,0
4,0	4,0	4,0	4,0

(c)

1,0	2,0	5,0	0,0
2,0	3,0	6,0	1,0
4,0	5,0	8,0	3,0
5,0	6,0	9,0	4,0

(d)

1,0	2,0	0,5	1,5
2,0	4,0	1,0	3,0
4,0	8,0	2,0	6,0
3,0	6,0	1,5	4,5

(e)

70	13	19	10
49	40	49	35
40	20	27	15
90	15	20	12

(f)

Figura 1 – Tipos de padrões exibidos por coclusters: (a) valores constantes, (b) valores constantes nas colunas, (c) valores constantes nas linhas, (d) valores coerentes (modelo aditivo), (e) valores coerentes (modelo multiplicativo) e (f) evolução coerente.

2.1.2 Estrutura em uma base de dados

Além dos diferentes padrões que podem definir um cocluster, a forma e distribuição dos coclusters na base de dados leva a diferentes abordagens para sua busca. A Figura 2 exibe as estruturas exploradas até hoje na literatura (MADEIRA; OLIVEIRA, 2004). O exemplo (a) indica a busca de um único cocluster base; já (b) representa a busca de múltiplos coclusters com linhas e colunas exclusivas; relaxando essa estrutura temos (d) e (e) que representam respectivamente a busca de coclusters com colunas ou linhas exclusivas, permitindo o compartilhamento de linhas ou colunas entre coclusters; (c) representa uma estrutura de tabuleiro de xadrez, na qual toda a matriz é particionada em coclusters alinhados; (f) representa coclusters com sobreposição seguindo uma estrutura hierárquica em árvore, de modo que o cocluster filho cinza escuro está totalmente contido no cocluster pai maior, representado pela união da cinza claro e cinza escuro; no exemplo (g) temos coclusters sem sobreposição e sem exclusividade de linhas ou coluna, os diferentes tons de cinza destacam diferentes coclusters; o exemplo (h) esquematiza coclusters com sobreposição em estrutura hierárquica, onde cada cocluster pode ter outro cocluster filho contido nele (marcados em cinza escuro), mas não há sobreposição entre diferentes linhas hierárquicas; por fim o exemplo (i) esquematiza coclusters com posicionamento e sobreposição arbitrários, onde as cores destacam diferentes coclusters. As principais estruturas exploradas na literatura são o tipo xadrez (Figura 2(c)) e de posicionamento arbitrário com sobreposição (Figura 2(i)), pois são comumente encontradas em bases reais.

Apesar de a Figura 2 induzir a crer que os coclusters são contíguos na matriz de dados, isso nem sempre é verdade, já que o cocluster é formado por um subconjunto de objetos e atributos, independentemente de sua posição original na matriz de dados. Logo, coclusters não contíguos são bastante comuns, particularmente para a estrutura da Figura 2(i).

2.2 Relação com análise de conceitos formais (FCA)

Análise de conceitos formais (FCA, do inglês *Formal Concept Analysis*) é um método de análise de dados com aplicação em diversos domínios (GANTER *et al.*, 2005). A definição de um conceito formal é análoga à de um cocluster de valores constantes, se considerada uma

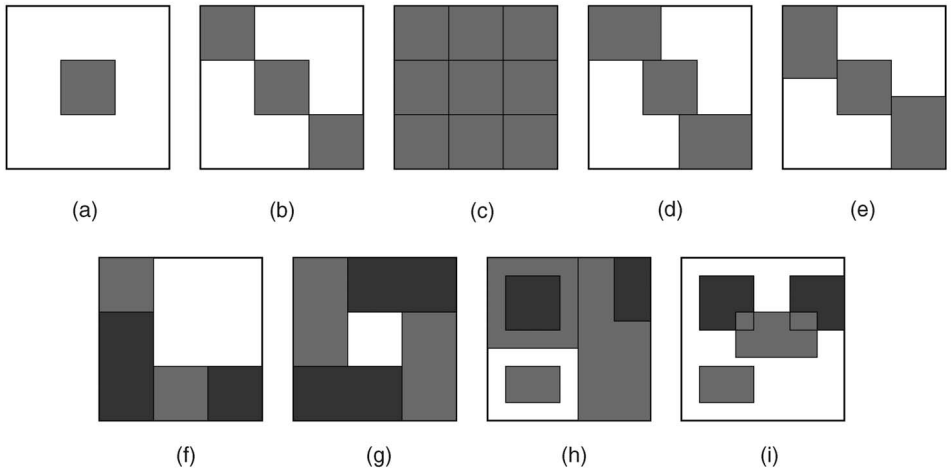


Figura 2 – Estruturas de coclusters em uma base de dados (MADEIRA; OLIVEIRA, 2004): (a) cocluster único, (b) coclusters com linhas e colunas exclusivas, (c) Estrutura em xadrez sem sopreposição, (d) coclusters com linhas exclusivas, (e) coclusters com colunas exclusivas, (f) coclusters sem sobreposição com estrutura em árvore, (g) coclusters sem sobreposição mas sem exclusividade de linhas ou colunas, (h) coclusters com sobreposição em estrutura hierárquica, (i) coclusters com posicionamento arbitrário e com sobreposição.

matriz binária. Se X é um conjunto de objetos e Y um conjunto de atributos, uma relação binária $M \subseteq X \times Y$ é denominada contexto formal $\mathscr{B}(X,Y,M)$. Dados $i \in X$ e $j \in Y$, iMj diz que o objeto i tem o atributo j. M pode ser representada por uma matriz binária $\{M_{ij}\}$, com $M_{ij} = 1 \Leftrightarrow iMj$. Para um subconjunto de objetos $I \subseteq X$ (ou de atributos $J \subseteq Y$), define-se o conjunto $I' \in Y$ de atributos comuns entre os objetos de I (ou conjunto $J' \in X$ de objetos comuns entre os atributos de J):

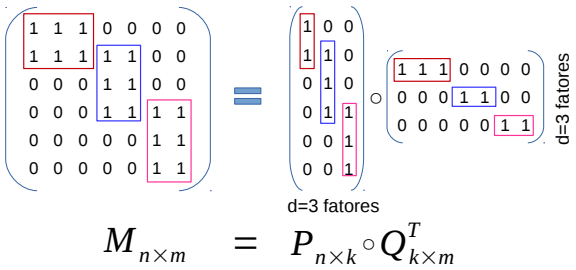
$$I' = \{j \in Y | \forall i \in I : iMj = 1\} \tag{2.1}$$

$$J' = \{i \in X | \forall j \in J : iMj = 1\} \tag{2.2}$$

(I,J) é um conceito formal se e somente se $I' = J$ e $J' = I$. I é chamado de *extent* e J é chamado de *intent*. O conjunto de todos os conceitos formais equivale ao contexto formal $\mathscr{B}(X,Y,M)$. Considerando a matriz M, o conceito formal (I,J) equivale ao cocluster (I,J), $I \in X$ e $J \in Y$. Cada combinação de linhas ou colunas de M pode gerar um conceito, de modo que o número de conceitos formais tende a crescer exponencialmente com o tamanho da matriz.

2.3 Fatoração booleana baseada em FCA

A fatoração booleana de matrizes (BMF, do inglês *Boolean matrix factorization*) é a decomposição da matriz $M \in \{0,1\}^{n \times m}$ em um produto matricial booleano $P \circ Q^T$ de matrizes binárias $P(n \times k)$ e $Q(m \times k)$. O produto booleano é definido de maneira análoga ao produto



$$M_{n\times m} = P_{n\times k} \circ Q^T_{k\times m}$$

Figura 3 – Exemplo explicitando a relação entre a BMF e a coclusterização binária. Cada fator em P e Q corresponde à um cocluster de M.

matricial tradicional, da seguinte forma:

$$(P \circ Q^T)_{ij} = \bigvee_{l-1}^{k} P_{il} \cdot Q_{jl}, \tag{2.3}$$

onde $\bigvee$ denota o "ou" lógico (disjunção) e $\cdot$ denota o "e" lógico (conjunção).

Belohlavek e Vychodil (2010) propuseram um método de fatoração de matrizes binárias baseado em análise de conceitos formais. A relação entre os fatores que compõem as matrizes P e Q e os conceitos formais de M é da seguinte forma. Considere $\mathscr{F} \subseteq \mathscr{B}(X,Y,M)$ um subconjunto de todos os conceitos formais de M:

$$\mathscr{F} = \{(I_1,J_1),(I_2,J_2),\cdots,(I_k,J_k)\}, \tag{2.4}$$

Vamos definir as matrizes $P_{\mathscr{F}}$ e $Q_{\mathscr{F}}$:

$$P_{\mathscr{F}(il)} = \begin{cases} 1, & \text{se } i \in I_l, \\ 0, & \text{caso contrário,} \end{cases} \qquad Q_{\mathscr{F}(jl)} = \begin{cases} 1, & \text{se } j \in J_l, \\ 0, & \text{caso contrário,} \end{cases} \tag{2.5}$$

para $l = \{1,2,\cdots,k\}$. Cada par de linhas de $P_{\mathscr{F}}$ e $Q^T_{\mathscr{F}}$ corresponde a um conceito formal de $\mathscr{F}$. Na Figura 3 temos um exemplo de fatoração, no qual as cores explicitam essa relação entre fatores de $P_{\mathscr{F}}$ e $Q^T_{\mathscr{F}}$ e coclusters da matriz M.

Belohlavek e Vychodil (2010) demonstraram que para toda matriz binária M existe um conjunto de conceitos formais $\mathscr{F} \subseteq \mathscr{B}(X,Y,M)$ tal que $M = P_{\mathscr{F}} \circ Q^T_{\mathscr{F}}$. Além disso, sendo $M = P \circ Q^T$, onde P é $m \times k$ e Q é $n \times k$, o número ótimo de fatores k é a cardinalidade deste conjunto $\mathscr{F}$, de modo que $|\mathscr{F}| \leq k$.

Dois algoritmos foram propostos por Belohlavek e Vychodil (2010). O primeiro se baseia na busca de todos os conceitos formais de M para posterior determinação do subconjunto $\mathscr{F}$, enquanto o segundo determina incrementalmente os conceitos formais necessários para determinar a matriz M, de maneira gulosa, visando a cada passo o conceito que cobre o máximo número de 1's na matriz. Na média, o segundo algoritmo — utilizado nesse trabalho

— acha uma quantidade de fatores próxima à do primeiro algoritmo, sendo mais eficiente, com complexidade de $O(kmn)$.

Seu pseudocódigo é apresentado no Algoritmo 1. Na linha 2 é inicializada a estrutura que controla quais 1's faltam ser cobertos. Na linha 3 inicializa-se o conjunto que guarda os fatores que vão sendo encontrados. Segue-se um ciclo (linhas 4 a 17) que busca os fatores que é repetido até que toda a matriz esteja coberta. Em cada ciclo, parte-se de um conjunto de atributos vazio (linha 5) e itera-se sobre os atributos, incluindo sempre aquele que gera o conceito de maior cobertura (linhas 8 a 10). Isso é feito enquanto existir um atributo a incluir que aumente a cobertura do conceito que está sendo construído (linha 7). Ao final de cada ciclo é gerado um conceito que é armazenado, e atualizado o controle dos 1's que já foram cobertos (linhas 12 a 16). A busca para quando todos os 1's são cobertos, retornando o conjunto dos conceitos encontrados, que correspondem aos fatores da BMF.

Algoritmo 1: Algoritmo guloso para obtenção da BMF. A cada etapa, são buscados conceitos que maximizem $J \oplus j = ((J \cup \{j\})' \times (J \cup \{j\})'') \cap \mathscr{M}$. O operador $|\ |$ indica a cardinalidade de um conjunto.

Entrada: M: matriz binária de objetos por atributos
Saída: $\mathscr{F}$: conjunto de fatores

1 **início**
2 $\quad \mathscr{M} \longleftarrow \{\langle i, j\rangle | M_{ij} = 1\}$
3 $\quad \mathscr{F} \longleftarrow \emptyset$
4 $\quad$ **enquanto** $\mathscr{M} \neq \emptyset$ **faça**
5 $\quad\quad J \longleftarrow \emptyset$ `/* Conjunto de atributos */`
6 $\quad\quad c \longleftarrow 0$ `/* Cobertura atual */`
7 $\quad\quad$ **enquanto** *existe atributo* $j \notin J$ *tal que* $|J \oplus j| > c$ **faça**
8 $\quad\quad\quad$ selecionar atributo $j \notin J$ que maximiza $J \oplus j$
9 $\quad\quad\quad I \longleftarrow (J \cup j)''$
10 $\quad\quad\quad c \longleftarrow |J' \times J \cap \mathscr{M}|$
11 $\quad\quad$ **fim**
12 $\quad\quad I \longleftarrow J'$
13 $\quad\quad$ adicionar $\langle I, J\rangle$ à $\mathscr{F}$
14 $\quad\quad$ **para cada** $\langle i, j\rangle \in I \times J$ **faça**
15 $\quad\quad\quad$ remover $\langle i, j\rangle$ de $\mathscr{M}$
16 $\quad\quad$ **fim**
17 $\quad$ **fim**
18 $\quad$ **retorna** $\mathscr{F}$
19 **fim**

É possível parar a busca de conceitos antes que a matriz esteja completamente coberta, ficando com um número de fatores $l < k$. Obtém-se assim uma aproximação de M. A escolha adequada de l não é muito intuitiva, mas l pode ser determinado através da definição de um limite mínimo de cobertura da matriz original M.

2.4 Coclusterização contígua em detecção de padrões em séries temporais

Coclusters com evolução coerente são particularmente interessantes para análise de séries temporais. Juntando um grupo de séries temporais simultâneas, obtém-se uma matriz cujas linhas são séries e cujas colunas são instantes de tempo. Nessa matriz, esse tipo de cocluster pode identificar os intervalos de tempo nos quais essas séries apresentaram um comportamento coerente, por exemplo, subindo e descendo nos mesmos instantes. Esse tipo de análise já foi aplicado para identificar padrões na expressão dinâmica de genes, indicando um potencial uso para análise de outras séries temporais biológicas (MADEIRA *et al.*, 2010).

Nesse cenário, exigir uma estrutura de colunas contíguas é um relaxamento razoável para o problema da enumeração de coclusters com posição arbitrária e sobreposição. Assim, serão retornados coclusters que correspondem a intervalos de tempo contíguos. Madeira *et al.* (2010) propuseram um algoritmo capaz de enumerar todos os coclusters de colunas contíguas (CCC) maximais, com complexidade linear no número de entradas da matriz.

O algoritmo trabalha numa versão previamente normalizada, diferenciada e quantizada das séries. Na proposta, a série diferenciada é quantizada em 3 símbolos: Subindo (U, *up*), descendo (D, *down*) ou não varia (N, *no change*). A Figura 4 mostra um exemplo com os coclusters encontrados após esse processamento. A regra de diferenciação é a seguinte:

$$M'_{ij} = \begin{cases} \frac{M_{i(j+1)} - M_{ij}}{M_{ij}}, & \text{se } M_{ij} \neq 0, \\ -1, & \text{se } M_{ij} = 0 \text{ e } M_{i(j+1)} < 0, \\ +1, & \text{se } M_{ij} = 0 \text{ e } M_{i(j+1)} > 0, \\ 0, & \text{se } M_{ij} = 0 \text{ e } M_{i(j+1)} = 0, \end{cases} \tag{2.6}$$

onde M'_{ij} é a entrada da i-ésima linha e j-ésima coluna da nova matriz diferenciada M'. A regra de quantização é a seguinte:

$$M''_{ij} = \begin{cases} D, & \text{se } M'_{ij} \leq t, \\ U, & \text{se } M'_{ij} \geq t, \\ N, & \text{caso contrário,} \end{cases} \tag{2.7}$$

onde M''_{ij} é a entrada da i-ésima linha e j-ésima coluna da nova matriz quantizada M''. Em seguida, a cada símbolo é adicionado o número da coluna onde ele ocorreu, além de um símbolo de terminação que identifica cada série (ex: \$1, \$2, ..., \$n). A Figura 5a mostra o exemplo da Figura 4 após esse tratamento. A sequência de símbolos de cada linha forma uma *string* que representa a série correspondente. O conjunto de todas essas strings é usado para a construção de uma árvore de sufixos generalizada (UKKONEN, 1995).

Madeira *et al.* (2010) mostram que, baseando-se nessa árvore de sufixos, é possível encontrar todos os coclusters contíguos maximais. Cada nó interno da árvore equivale a um

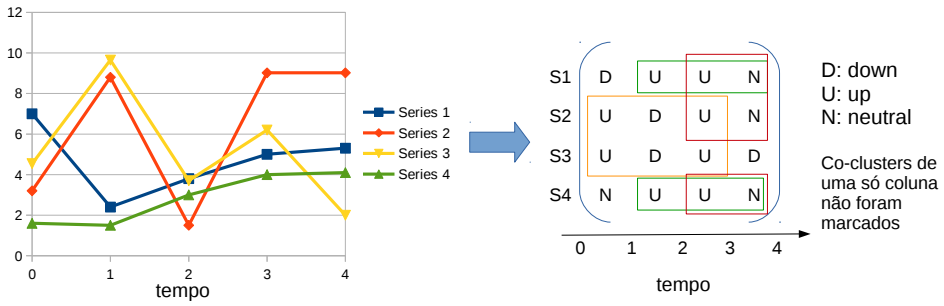


Figura 4 – Exemplo de coclusterização realizada pelo CCC-biclustering. As séries devem ser previamente diferenciadas e quantizadas.

cocluster, e ele será maximal sob certas condições. Na Figura 5, tem-se um exemplo de árvore de sufixos construída a partir do exemplo da Figura 4, na qual são destacados os nós internos equivalentes aos coclusters marcados.

Para uma dada *string*, o conjunto de seus sufixos é formado pela própria *string*, pela *substring* a partir da segunda posição até o fim, a *substring* a partir da terceira posição até o fim, e assim por diante, até chegar-se à *substring* com um único símbolo. Considerando a primeira linha do exemplo da Figura 5a, o conjunto de sufixos da *string* " D1 U2 U3 N4 $1" é {"D1 U2 U3 N4 $1", "U2 U3 N4 $1", "U3 N4 $1", "N4 $1", "$1" }.

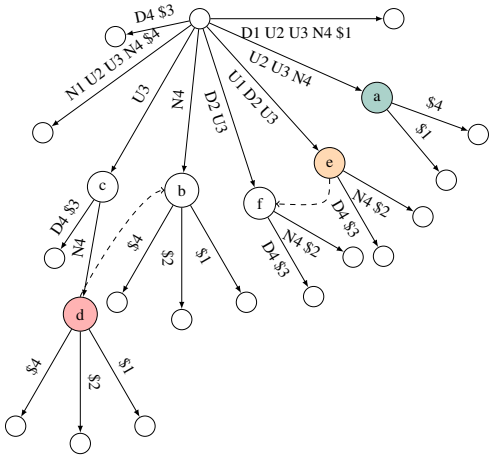
A construção da árvore se dá pela inserção sucessiva de todos os sufixos de uma *string* (para cada *string*, caso seja considerado um conjunto). De cada nó só pode sair uma única aresta iniciada por um certo símbolo. Começa-se com um nó raiz, do qual sai uma aresta com a *string* em questão (o primeiro sufixo de um *string* é a própria *string*). Em seguida, é inserido o segundo sufixo (a partir do segundo símbolo da *string* até o seu final), o terceiro (a partir do terceiro símbolo) e assim por diante. Por exemplo, após a inserção de toda a primeira *string* "D1 U2 U3 N4 $1" (Figura 5a), a árvore terá cinco nós filhos, e a aresta de cada um indica um dos sufixos da primeira *string*. Após terminar a primeira *string*, passa-se aos sufixos da segunda, e assim por diante.

Quando o primeiro símbolo do novo sufixo coincidir com um dos ramos já existentes na árvore, o novo sufixo será incluído criando um nó interno que marca uma bifurcação no ramo existente. A aresta até a bifurcação corresponde ao trecho de coincidência entre o novo e o antigo sufixo, e as arestas após a bifurcação indicam as diferentes terminações. Por exemplo, quando o terceiro sufixo da segunda *string*, "U3 N4 $2", vai ser inserido na árvore, já vai existir um ramo começando com "U3" proveniente do terceiro sufixo da primeira *string*, "U3 N4 $1". Torna-se necessário criar uma bifurcação, criando o nó (d) (Figura 5b). Note que o nó interno é criado justamente quando há coincidência de sufixos entre séries diferentes.

Eventualmente são criados nós internos cujas sub-árvores são idênticas à de outro nó interno criando anteriormente. Tomando-se o exemplo da Figura 5, ao se inserir o quarto sufixo da segunda *string*, "N4 $2", há uma coincidência de símbolo inicial com o quarto sufixo

D1	U2	U3	N4	$1
U1	D2	U3	N4	$2
U1	D2	U3	D4	$3
N1	U2	U3	N4	$4

(a) Matriz após adição de número de coluna e marcadores de terminação das *strings*.



(b) Árvore de sufixos correspondente.

Figura 5 – A geração da árvore de sufixos permite identificar os coclusters. Os nós (a), (d) e (c) indicam os coclusters marcados na Figura 4. As arestas tracejadas representam ligações de sufixos, que não fazem parte da árvore, e indicam nós com a mesma sub-árvore. Coclusters com apenas uma coluna não foram marcados.

da primeira *string*, “N4 $1”. Isso leva à criação do nó (b), que contém a mesma sub-árvore do nó (d). Por isso, é adicionada uma ligação de sufixo de (d) para (b), que indica que esses nós têm a mesma sub-árvore. As ligações de sufixo não são ramos da árvore, e estão indicadas por setas pontilhadas na Figura 5b.

Desse modo, os nós internos da árvore indicam quando um sufixo — indicado pelo caminho da raiz até o dado nó — se repete. No contexto do algoritmo CCC-biclustering, as repetições de sufixos ocorrem quando diferentes séries apresentarem o mesmo padrão de símbolos em colunas correspondentes — formando um cocluster. Por exemplo, na Figura 5a, o sufixo “U2 U3 N4” se repete nas séries 1 e 4, o que é identificado pelo nó interno (a) da árvore (Figura 5b).

O Algoritmo 2 descreve como a árvore é percorrida a fim de determinar quais nós equivalem a coclusters maximais. Primeiro a árvore é percorrida para calcular a profundidade de cada nó interno, e também o número de folhas da sub-árvore que começa nele (linhas 3 a 7). Nessa passada todos os nós são marcados como potenciais coclusters (linha 4). Em seguida a árvore é novamente percorrida verificando-se as ligações de sufixo: os nós que recebem uma ligação e têm o mesmo número de folhas que o nó de origem são desconsiderados, pois não representam coclusters maximais (linhas 8 a 11). Por fim, todos os nós internos que restaram são retornados como coclusters (linhas 13 a 18). Esse algoritmo foi utilizado na proposta apresentada no capítulo 8.

Algoritmo 2: Algoritmo CCC-biclustering, conforme Madeira *et al.* (2010). A profundidade $P(v)$ de um nó v é a quantidade de símbolos existentes do nó raiz até o nó v.

Entrada: $S = S_1, \cdots, S_N$: conjunto de strings representando as séries temporais
Saída: $\mathscr{B}$: conjunto de ccc-biclusters maximais

```
início
    Construir uma árvore generalizada de sufixos T para strings em S
    para cada nó interno v ∈ T faça
        Marcar v como válido
        Calcular profundidade P(v) do nó v
        Calcular o número de folhas L(v) da sub-árvore de raiz v
    fim
    para cada nó interno v ∈ T faça
        se existir uma ligação de sufixo partindo de v para um nó u e L(u) == L(v) então
            Marcar nó u como inválido
        fim
    fim
    para cada nó interno v ∈ T faça
        se v está marcado com válido então
            B ← v
        fim
    fim
    retorna B
fim
```

2.5 Coclusterização na neurociência

Existem algumas aplicações de algoritmos de coclusterização na neurociência, mas ainda é um campo de aplicação pouco explorado. Até onde se pode verificar, não existem ainda aplicações de coclusterização diretamente em séries temporais de fMRI (ressonância magnética funcional, do inglês *functional magnetic resonance imaging)* cerebral, nem como método de obtenção de mapas funcionais cerebrais.

Busygin *et al.* (2007) utilizaram uma técnica de biclusterização na análise de dados de EEG extra-cranial de pacientes de epilepsia que estavam sendo tratados com estimulação vaga de nervos (Vagus nerve stimulation – VNS). O objetivo era extrair um marcador fisiológico que permitisse a otimização dos parâmetros da VNS, como por exemplo frequência e amplitude da estimulação.

Lu *et al.* (2013) e Lu *et al.* (2014) usaram coclusterização harmônica para analisar a morfologia das árvores dendríticas de diferentes tipos de células neurais. São analisadas imagens de microscopia de células cerebrais de diversos tipos, como neurônios e células micro-glia, extraindo 127 atributos para cada célula. Por fim é aplicada a coclusterização para identificar grupos de células semelhantes, identificando padrões morfológicos significativos.

Lin *et al.* (2010) utilizaram uma técnica de coclusterização para analisar dados de

ressonância magnética de tensor de difusão (DT-MRI), a fim de realizar a segmentação dos voxels corticais em grupos segundo os padrões observados de conectividade anatômica entre eles. Contudo, a formulação de coclusterização proposta difere do que foi apresentado neste trabalho. Nesse caso, a entrada do algoritmo é um grafo com arestas ponderadas, cujos nós representam diferentes voxels da imagem cortical, e as arestas representam conexões entre eles. O objetivo é descobrir pares de grupos — denominados coclusters — com grande conectividade entre eles, e baixa conectividade com outros grupos. Desse modo, o método permite a segmentação anatômica considerando ligações intra-corticais. Lin *et al.* (2007) e Lin *et al.* (2008) apresentam linhas de trabalho semelhantes.

Nesse trabalho, a coclusterização contígua apresentada na seção 2.4 será utilizada no contexto de análise de séries temporais de atividade cerebral. Essa frente de trabalho será detalhada no capítulo 8.

3 Sistemas de recomendação

Com a crescente quantidade de informação e opções de consumo disponíveis, torna-se cada vez mais necessária uma ferramenta que auxilie a discernir o que é realmente relevante. Esse é o objetivo principal dos sistemas de recomendação: sugerir a usuários itens que possam lhes interessar. A denominação "item" denota genericamente o tipo de produto ou serviço recomendado pelo sistema aos usuários, que varia segundo o domínio de aplicação.

Em sua forma mais simples, recomendações personalizadas são feitas na forma de listas de itens, ordenados pelo potencial interesse ao usuário. Para tanto, o sistema deve tentar prever os produtos ou serviços de maior relevância, baseando-se em dados disponíveis sobre o usuário, como preferências e restrições, avaliações de itens e histórico de navegação (RICCI *et al.*, 2011).

Sistemas de recomendação surgiram como uma área de pesquisa independente em meados dos anos 1990, e continuam despertando crescente interesse. Pode-se dizer que seu estudo é relativamente recente se comparado a outras ferramentas clássicas de sistemas de informação, como bancos de dados ou motores de busca (RICCI *et al.*, 2011). Sistemas de recomendação têm sido aplicados em diversos domínios, recomendando filmes, músicas, livros, produtos eletrônicos, páginas web, notícias, amigos em redes sociais e até mesmo serviços financeiros (JANNACH *et al.*, 2010).

De maneira geral, sistemas de recomendação se baseiam em duas principais estratégias: filtragem baseada em conteúdo e filtragem colaborativa. Esses tipos de sistemas de recomendação são descritos a seguir, com maior foco na filtragem colaborativa, de maior relevância para este trabalho.

3.1 Filtragem baseada em conteúdo

Na filtragem baseada em conteúdo, o sistema recomenda um item com base em sua descrição e com base em um perfil de interesses do usuário. Para tanto, é necessário que cada usuário e cada item tenha um perfil que o caracterize, com atributos relevantes.

A descrição de um item depende diretamente do domínio de aplicação. Por exemplo, um perfil de um filme pode incluir atributos como gênero, diretor e atores principais. A similaridade dos itens é calculada com base nos atributos associados aos itens comparados (RICCI *et al.*, 2011).

Perfis de usuário podem conter informações demográficas, respostas dadas a um questionário, avaliações sobre itens e comentários feitos pelo usuário. Neste caso, é necessário o fornecimento explícito de informações, com participação ativa do usuário. Também pode-

se obter informações de maneira implícita, baseando-se em ações passadas do usuário, por exemplo, procurando atributos em comum nos filmes de que o usuário gostou e entre os de que ele não gostou (VERONEZE, 2011).

Os perfis permitem a associação de usuários com produtos adequados. A filtragem baseada em conteúdo depende então da coleta de informação extra, que pode não ser viável. Esta estratégia de filtragem é comumente usada como acessório à filtragem colaborativa, por ser capaz de produzir recomendações em casos difíceis para este último método, como para um usuário que avaliou poucos itens, mas informou suas preferências (CANDILLIER *et al.*, 2009).

3.2 Filtragem colaborativa

Na filtragem colaborativa, as recomendações se baseiam apenas em registros do comportamento do usuário em relação aos itens. Por exemplo, itens que foram comprados, itens observados, notas dadas a filmes e tempo de permanência numa página web. A partir disso, a filtragem colaborativa procura extrair relações entre usuários e interdependências entre itens de modo a identificar novas associações usuário-item (KOREN *et al.*, 2009).

Para um dado tipo de registro, pode-se organizar os dados em uma tabela (matriz) de usuários por itens, cujas entradas contêm um valor numérico que quantifica a interação medida (a nota dada ao item, por exemplo). Na filtragem colaborativa, existem duas principais abordagens para relacionar itens e usuários: métodos baseados em vizinhança (similaridade de itens e/ou usuários) e métodos baseados em modelo (KOREN; BELL, 2011). A seguir, serão detalhadas essas duas categorias e seus principais métodos. Na descrição dos algoritmos, U denota um conjunto de usuários e I um conjunto de itens, e r_{ui} a avaliação que um usuário $u \in U$ atribui a um item $i \in I$.

3.2.1 Métodos baseados em vizinhança

Os métodos baseados em vizinhança focam nas relações entre usuários ou entre itens. Eles são os principais componentes de uma classe ainda mais ampla, denominada tradicionalmente de métodos baseados em memória (ADOMAVICIUS; TUZHILIN, 2005; CANDILLIER *et al.*, 2009; REN *et al.*, 2014).

3.2.1.1 Abordagem baseada em usuário

Na abordagem baseada em usuário, o sistema identifica usuários vizinhos (que apresentam preferências similares), recomendando itens bem cotados entre eles (MCLAUGHLIN; HERLOCKER, 2004). Espera-se que usuários que têm muito em comum tendam a gostar dos mesmos itens. Sendo $N(u,i)$ os vizinhos mais próximos de u que avaliaram o item i, uma nota

desconhecida $\hat{r}_{ui}$ é predita da seguinte forma:

$$\hat{r}_{ui} = \frac{\sum_{j \in N(u,i)} sim(u,j) r_{ji}}{\sum_{j \in N(u,i)} sim(u,j)} \tag{3.1}$$

onde $sim(u,j)$ é a similaridade entre o usuário u e o usuário j.

Note que nem todos os usuários utilizam a escala da mesma forma. Para um deles, uma nota 3 significa uma preferencia considerável, enquanto para outro a mesma nota pode significar que o item é irrelevante. Para levar em consideração esse efeito pode-se prever a nota baseada nos desvios da média de cada usuário, da seguinte forma:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{j \in N(u,i)} sim(u,j)(r_{ji} - \bar{r}_j)}{\sum_{j \in N(u,i)} sim(u,j)} \tag{3.2}$$

onde $\bar{r}_u$ é a média das notas dadas pelo usuário u.

3.2.1.2 Abordagem baseada em item

Uma alternativa é a abordagem baseada em item, na qual estima-se a preferência do usuário por um item com base em suas interações com itens vizinhos (SARWAR *et al.*, 2001). Neste caso, itens são considerados vizinhos se apresentam padrões similares de interação com os usuários (itens comprados por um mesmo grupo de usuários, por exemplo). Espera-se que um dado usuário avalie itens vizinhos da mesma forma.

Sendo $N(i,u)$ os vizinhos mais próximos de i que avaliaram o item u, uma nota desconhecida é predita da seguinte forma:

$$\hat{r}_{ui} = \frac{\sum_{j \in N(i,u)} sim(i,j) r_{uj}}{\sum_{j \in N(i,u)} sim(i,j)} \tag{3.3}$$

onde $sim(i,j)$ é a similaridade entre o item i e o item j.

3.2.1.3 Abordagem baseada em usuário e item

As abordagens baseadas unicamente em usuário consideram a similaridade dos usuários ao longo de todos os itens da base. Analogamente, as abordagens baseadas em item consideram a similaridade dos itens ao longo de todos os usuários da base. Desse modo, essas técnicas não são capazes de detectar uma similaridade parcial, como um subgrupo de usuários que tem opiniões parecidas para um subgrupo de itens, por exemplo. Nesse contexto, a coclusterização pode ser uma ferramenta útil na detecção de tais padrões locais.

Uma das primeiras propostas nesse sentido considera uma versão discretizada da matriz de avaliações na busca por coclusters, e gera uma lista de recomendação com os itens mais frequentes nos k coclusters mais próximos do usuário alvo (SYMEONIDIS *et al.*, 2006) (SYMEONIDIS *et al.*, 2007). Essa abordagem é baseada no cálculo prévio de todos os coclusters da

matriz, o que pode ser muito custoso para bases de dados grandes. Além disso, atualizações da base exigem que todo esse cálculo seja refeito.

Desde então diversos outros trabalhos procuraram usar a coclusterização em filtragem colaborativa (BOUCHER-RYAN; BRIDGE, 2006; XU *et al.*, 2012; KASHNITSKY; IGNATOV, 2014; SAITO; OKADA, 2014). Buscando melhorar alguns aspectos da escalabilidade, Yokoyama e Okada (2014) propõem um novo método com duas características principais. Primeiro, a busca de coclusters se dá considerando apenas os coclusters que incluem itens alvo da consulta, reduzindo o espaço de busca. Segundo, o método de coclusterização utilizado não necessita re-enumerar todos os coclusters para a base inteira em caso de atualizações dos dados.

Usando conceitos de FCA (análise de conceitos formais, do inglês *formal concept analysis*), Alqadah *et al.* (2014) propõem uma abordagem escalável e *online*. Primeiro o cocluster mínimo que contém o usuário alvo é inferido *online*: trata-se do subconjunto de itens com os quais esse usuário interagiu, e do subgrupo de usuários que interagiu conjuntamente com esse subgrupo de itens. Em seguida, é explorada a vizinhança desse cocluster, que levará aos itens candidatos à recomendação. Essa abordagem não necessita o prévio cálculo de todos os coclusters, sendo mais escalável.

3.2.1.4 Métricas de similaridade

É evidente que a escolha da métrica de similaridade influencia diretamente a predição. Candillier *et al.* (2009) realizaram uma comparação destes métodos utilizando diferentes métricas, como Jaccard, correlação de Pearson e cosseno, com diversos ajustes e pesos. Para o método baseado em usuário, a similaridade de melhor desempenho foi correlação de Pearson com pesos da similaridade Jaccard, que consiste no produto das duas métricas. Para o método baseado em item, a melhor métrica foi a similaridade do cosseno ajustado.

A seguir, são apresentadas algumas das principais métricas utilizadas em métodos baseados em vizinhança. $I(x)$ é o conjunto de itens avaliados por um usuário x e $U(x)$ o conjunto de usuários que avaliaram um item x. Ainda, $I(x,y) = I(x) \cap I(y)$ e $U(x,y) = U(x) \cap U(y)$. O operador $|\cdot|$ indica a cardinalidade do conjunto.

$$\text{Jaccard: } sim(x,y) = \frac{|I(x) \cap I(y)|}{|I(x) \cup I(y)|} \tag{3.4}$$

$$\text{Pearson: } sim(x,y) = \frac{\sum_{i \in I(x,y)} (r_{xi} - \bar{r}_x)(r_{yi} - \bar{r}_y)}{\sqrt{\sum_{i \in I(x,y)} (r_{xi} - \bar{r}_x)} \sqrt{\sum_{i \in I(x,y)} (r_{yi} - \bar{r}_y)^2}} \tag{3.5}$$

$$\text{Cosseno: } sim(x,y) = \frac{\sum_{u \in U(x,y)} (r_{ux} r_{uy})}{\sqrt{\sum_{u \in U(x,y)} r_{ux}^2} \sqrt{\sum_{u \in U(x,y)} r_{uy}^2}} \tag{3.6}$$

$$\text{Cosseno ajustado: } sim(x,y) = \frac{\sum_{u \in U(x,y)} (r_{ux} - \bar{r}_u)(r_{uy} - \bar{r}_u)}{\sqrt{\sum_{u \in U(x,y)} (r_{ux} - \bar{r}_u)^2} \sqrt{\sum_{u \in U(x,y)} (r_{uy} - \bar{r}_u)^2}} \tag{3.7}$$

3.2.2 Métodos baseados em modelo

Nos métodos baseados em modelo, a ideia é derivar um modelo dos dados (*offline*) e utilizá-lo para predizer as avaliações *online*, o mais rápido possível (VERONEZE, 2011). Os primeiros modelos propostos consistiam em agrupar os usuários com alguma técnica de clusterização, e então predizer a avaliação de um usuário para um item considerando apenas as avaliações dos usuários no mesmo grupo (CANDILLIER *et al.*, 2009).

Mais recentemente, modelos de fatores latentes se mostraram promissores para melhorar a acurácia das recomendações, especialmente após a competição Netflix em 2009, na qual métodos avançados de fatoração de matrizes foram utilizados por vários times na competição (JANNACH *et al.*, 2010). Modelos inspirados na abordagem de Koren e Bell (2011), que está baseada em decomposição de valores singulares (SVD, do inglês *singular value decomposition*), são os mais utilizados. Outras técnicas, como a fatoração não negativa (NMF, do inglês *non-negative matrix factorization*) e a fatoração booleana (BMF, do inglês boolean matrix factorization), têm sido menos exploradas nesse contexto (IGNATOV *et al.*, 2014).

3.2.2.1 Fatoração SVD

Um método muito utilizado em diversos domínios para identificar fatores latentes é o SVD. Contudo, ele não pode lidar com o problema de dados faltantes, muito comum em sistemas de recomendação, uma vez que é improvável que todos os usuários tenham avaliado todos os itens. Portanto, é necessário aproximar a fatoração levando em conta apenas as notas conhecidas.

Dada uma matriz de avaliações $M \in \mathbb{R}^{|U| \times |I|}$ onde U é o conjunto de usuários, I o de itens, e $|U|$ e $|I|$ suas cardinalidades, a fatoração de M em duas matrizes $P \in \mathbb{R}^{|U| \times d}$ e $Q \in \mathbb{R}^{|I| \times d}$ gera uma representação em um espaço de fatores latentes de dimensão d. Cada linha destas matrizes P e Q é um vetor (transposto) que representa um usuário e um item, respectivamente. No modelo mais simples, para um usuário u representado pela linha p_u de P, e um item não avaliado i representado pela linha q_i de Q, a nota de u para i pode ser predita pelo produto interno entre esses vetores:

$$\hat{r}_{ui} = p_u^T q_i = q_i^T p_u \tag{3.8}$$

Nesse caso, sendo R o conjunto de avaliações em M, a fatoração é obtida pela minimização do

erro quadrático regularizado:

$$\min_{P,Q} \sum_{r_{ui} \in R} \left(r_{ui} - p_u^T q_i\right)^2 + \lambda \left(\|q_i\|^2 + \|p_i\|^2\right) \tag{3.9}$$

Koren *et al.* (2009) propõem também um modelo mais elaborado, também inspirado em SVD, que leva em consideração a média geral de notas μ, a média de cada usuário b_u e a média de cada item b_i. Nesse caso a nota r_{ui} é dada por:

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u. \tag{3.10}$$

O problema de otimização regularizado correspondente fica:

$$\min_{P,Q,b_*} \sum_{u \in U} \sum_{i \in I} \nexists_{ui} (r_{ui} - p_u^T q_i)^2 + \lambda (b_i^2 + b_u^2 + \|q_i\|^2 + \|p_i\|^2), \tag{3.11}$$

onde $\nexists_{ui}$ é uma função indicadora que assume o valor 1 quando a avaliação está presente e zero quando a avaliação está ausente. Comumente, é utilizado o método do gradiente estocástico ou o método de quadrados mínimos alternado para resolver este problema. As constantes de regularização em geral são determinadas por validação cruzada (KOREN *et al.*, 2009).

Outro modelo proposto por eles, o SVD++, considera também quais itens foram avaliados por cada usuário, levando a uma melhor performance que o modelo anterior. Existem ainda modelos que levam em consideração quando cada avaliação foi feita, dentre outros fatores temporais, também visando melhor acurácia. Koren e Bell (2011) apresentam uma revisão abrangente destes métodos.

3.2.2.2 Fatoração booleana (BMF)

Ignatov *et al.* (2014) propuseram um modelo de recomendação baseado em uma fatoração booleana (ver seção 2.3). Nesse caso, M deve ser binária (ou binarizada segundo um limiar). Usuários e itens são representados por vetores binários. Após obter a fatoração binária, eles procedem com uma recomendação baseada em vizinhança do usuário, utilizando os vetores p_u para calcular a vizinhança de cada usuário u. Foi reportada uma performance comparável com o modelo SVD. Essa abordagem será estendida neste trabalho.

3.3 Comitês de filtros colaborativos

A ideia central da metodologia de comitês (*ensembles*) é combinar um conjunto de modelos que resolvem uma mesma tarefa a fim de obter uma solução final de melhor desempenho e/ou mais confiável do que as soluções individuais, com maior capacidade de generalização. Entretanto, para que essa melhora ocorra, os modelos de base devem apresentar diversidade no erro, ou seja, cometer erros diferentes. Assim, o erro de um modelo pode ser compensado pelo acerto de um outro (COELHO, 2006).

No contexto de filtragem colaborativa, diversas abordagens experimentais demonstraram esse efeito de melhoria devida ao *ensemble*. Koren (2009), por exemplo, usaram uma combinação de 107 modelos diferentes na solução vencedora do *Netflix challenge*. Os resultados foram combinados através de uma regressão linear. Nessa proposta, foram utilizados algoritmos diferentes para gerar os modelos-base, garantindo diversidade. De fato, seus resultados mostraram ser melhor combinar modelos bastante diferentes do que refinar um modelo em particular. Na competição KDD de 2011, Jahrer e Töscher (2011) também usam uma série de modelos diferentes, combinados através de uma rede neural.

Duas propostas se baseiam no algoritmo de *ensemble* AdaBoost, de Freund e Schapire (1995). O algorimo AdaBoost, e suas variantes, é um método de *ensemble* eficaz em melhorar os resultados de um dado modelo — de classificação ou regressão. Seu princípio básico é, a cada iteração, focar nas amostras que são mais difíceis de predizer. O próprio Freund e colegas propõem um algoritmo que gera recomendações na forma de rankings (FREUND *et al.*, 2003) . O algoritmo combina vários rankings "fracos" (baixa correlação com o ranking ideal) em um ranking final, gerando uma lista de recomendações ordenada segundo a relevância nesse ranking. Schclar *et al.* (2009) propõem um *ensemble* homogêneo, utilizando por base um modelo de regressão para predição das notas. Os modelos são combinados através de um regressor AdaBoost.RT.

Alguns trabalhos combinam diferentes modelos de fatoração de matrizes. DeCoste (2006) explora diferentes maneiras de combinar modelos baseados em fatoração de matriz com margem máxima (MMMF), obtidos com diferentes inicializações. As notas preditas por cada modelo são combinadas com uma média aritmética, e também com a moda (nota mais frequente). Wu (2007) combina três tipos de modelos de fatoração de matriz: a fatoração simples, a fatoração não-negativa e a MMMF. A diversidade surge do uso de diferentes parâmetros nas fatorações, como regularização e número de fatores. A predição final também é a média das predições individuais. Numa outra linha, Chen *et al.* (2015) agrega os resultados de múltiplas fatorações, obtidas a partir de submatrizes da matriz original. As submatrizes são determinadas via uma coclusterização que divide a matriz original em uma estrutura de xadrez (submatrizes alinhadas e sem sobreposição), reduzindo o custo computacional de cada fatoração individual.

Neste trabalho, será proposto um *ensemble* de preditores de notas, também baseados em um tipo de fatoração de matriz — a fatoração booleana. Será empregada uma técnica de projeções aleatórias a fim de introduzir diversidade para a geração de um ensemble baseado em fatoração booleana de matrizes. Cada componente do comitê gerará uma predição e as múltiplas predições serão agrupadas com uma regressão com regularização *elastic net*. Essa frente de trabalho será detalhada no capítulo 7.

4 Técnicas aproximadas para *k*-vizinhos mais próximos

Encontrar os k vizinhos mais próximos de uma dada entrada em uma base de dados é um problema recorrente em diversas áreas, como aprendizado de máquina, mineração de dados, processamento de imagens e visão computacional. O problema de busca dos k vizinhos mais próximos (kNN – k-*Nearest Neighbors*) consiste em encontrar os k elementos mais próximos do elemento de consulta, em um determinado conjunto de dados, considerando uma certa métrica de similaridade.

Análogo, mas ligeiramente diferente, é o problema de construção de um grafo kNN. Neste grafo, as entradas da base de dados são os nós, e de cada um saem k arestas (direcionadas) conectando a seus vizinhos mais próximos. Note que um problema pode ser convertido no outro sob certas condições, mas essa nem sempre é a melhor maneira de resolvê-los. Um grafo kNN pode ser utilizado para resolver o problema de busca kNN, mas somente quando todos os possíveis pontos de consulta são conhecidos no momento do treinamento. A busca kNN trabalha *online* e não assume essa condição. Um grafo kNN pode ser obtido através da execução de buscas kNN para cada elemento da base. Porém isso pode ser bastante custoso, pois como na busca kNN o foco é reduzir o tempo de consulta, no treinamento são criadas estruturas complexas para indexação (ZHANG *et al.*, 2013).

De todo modo, ambos os problemas apresentam complexidade alta quando a dimensão e a quantidade de amostras da base de dados cresce. Considerando n amostras num espaço de dimensão D, encontrar os kNN para cada ponto necessita de $(n-1)$ comparações, o que leva a uma complexidade de $O(Dn^2)$. Existem diversos outros métodos que são eficientes quando D é pequeno (1 ou 2), como diagramas de Voronoi, ou moderado (da ordem de dezenas), como *metric-trees* (LIU *et al.*, 2004). Todavia, esses métodos continuam custosos em grandes dimensões, o que motivou o desenvolvimento de técnicas aproximadas para kNN.

4.1 Busca kNN aproximada com LSH

Um dos principais métodos propostos para a busca kNN aproximada se baseia no uso de funções de *hashing* sensíveis à localidade (LSH - *Locally Sensitive Hashing*), proposto inicialmente por Indyk e Motwani (1998). A idéia central do LSH é aplicar várias funções de *hashing* de maneira que a probabilidade de colisão é muito maior para pontos próximos entre si do que para pontos distantes. Deste modo, para encontrar vizinhos do ponto de consulta, basta aplicar as funções de *hashing* e procurar no grupo com mesmo *hash*.

4.1.1 Famílias de funções LSH

Considere uma família de funções de *hashing* $\mathscr{H} : \mathbb{R}^d \to U$. Para quaisquer dois pontos $p, q \in \mathbb{R}^d$, suponha que seja escolhida com probabilidade uniforme uma função $h \in \mathscr{H}$. A família $\mathscr{H}$ é dita (R, cR, P_1, P_2)-sensível caso, para quaisquer $p, q \in \mathbb{R}^d$, $c > 1$, $R > 0$:

- se $\|p-q\| \leq R$ tem-se $Pr_{\mathscr{H}}[h(p) = h(q)] \geq P_1$ e
- se $\|p-q\| \geq cR$ tem-se $Pr_{\mathscr{H}}[h(p) = h(q)] \leq P_2$.

A Figura 6 procura ilustrar esse enunciado de maneira intuitiva. À esquerda, temos no centro o ponto q, e temos os pontos $p^{(1)}$ e $p^{(2)}$ a diferentes distâncias de q. À direita temos um gráfico que exemplifica como uma função LSH deve ser. Pontos como $p^{(1)}$, que estão a uma distância pequena de q, menor que R, terão uma probabilidade mínima garantida P_1 de ter o mesmo *hash* que q. Por outro lado, pontos como $p^{(2)}$, que estão distantes de q, mais do que cR, têm uma baixa probabilidade máxima garantida P_2 de ter o mesmo *hash* que q. Para pontos a uma distância de q entre R e cR, contudo, não há restrições impostas sobre a probabilidade de colisão de *hash*.

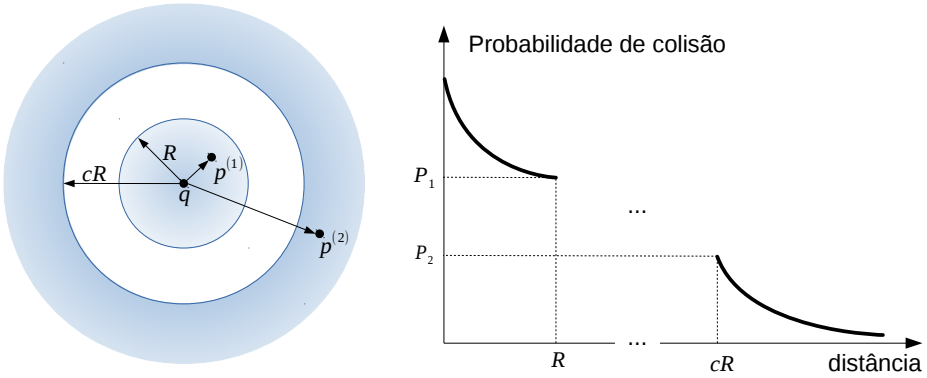


Figura 6 – Ilustração intuitiva do que se espera de uma função LSH. Pontos a uma distância menor que R devem ter probabilidade de colisão de no mínimo P_1, e pontos mais distantes que cR uma probabilidade de colisão menor que P_2. O gráfico à direita ilustra essa idéia. Note que nenhuma restrição é imposta para pontos situados a uma distância intermediária entre R e cR.

A escolha da família de funções de *hashing* vai depender da métrica de distância de interesse. Por exemplo, para a distância cosseno, a projeção de um vetor $p \in \mathbb{R}^d$ em um hiperplano aleatório $\{x \in \mathbb{R}^d : w^T x = 0\}$, $w \in \mathbb{R}^d$, pode definir um *hashing*. O sinal da projeção define uma função de *hash* com saída binária, da forma: $h_i(x) = sign(w_i^T x) \in \{-1, 1\}$. Dados dois pontos p e q com um ângulo θ entre eles (a uma distância $R = \cos\theta$), duas situações são possíveis: o hiperplano os separa com probabilidade $\frac{\theta}{180}$ e eles têm *hash* com sinais opostos; ou ambos estão do mesmo lado do hiperplano com chance $1 - \frac{\theta}{180}$ e eles têm *hash* de mesmo sinal. A probabilidade de colisão para pontos a uma distância R é portanto $1 - \frac{\cos^{-1} R}{180}$. Esse tipo de função define uma família de *hashing* $(R, cR, 1 - \frac{\cos^{-1} R}{180}, 1 - \frac{\cos^{-1} cR}{180})$-sensível. Outras métricas que também possuem uma família de funções LSH são Hamming, Jaccard, L1 e L2 (euclidiana) (LESKOVEC *et al.*, 2014, sec. 3.6-3.7).

No geral, são aplicadas várias funções de *hashing* da mesma família de modo a distanciar P_1 e P_2 tanto quanto necessário, influenciando também na margem c entre o raio de "proximidade" R e de "distância" cR (ANDONI; INDYK, 2008). São escolhidas aleatoriamente e com reposição k funções da família $\mathscr{H}$, para compor por concatenação uma função $g_j = (h_1, h_2, \cdots, h_k)$, que gera assinaturas para os pontos dados. Pontos com a mesma assinatura pertencem a um mesmo *bucket*.

Assim, se a probabilidade de colisão para pontos "distantes" for P_2 teremos uma probabilidade reduzida P_2^k, que tende a zero quanto maior for k. Entretanto um valor de k muito alto pode dificultar muito a colisão dos pontos "próximos". Para evitar esse problema, esse passo é repetido l vezes, gerando l tabelas de *hash* a partir das funções $g_j, j = 1, \cdots, l$. Se ocorrer colisão de *hash* em pelo menos uma tabela, os pontos são candidatos a vizinhos.

Cada tabela depende dos parâmetros k e l, que por sua vez dependem de R e do número de pontos n. Na prática, desde que l seja grande o suficiente, ele não interfere tanto na performance. Por outro lado, a sensibilidade ao parâmetro k é maior, o qual é bastante dependente dos dados e de n. Portanto as tabelas podem precisar ser reconstruídas caso haja uma mudança suficientemente grande da base de dados, para que a margem de precisão c seja mantida. Além disso, são necessárias muitas tabelas para garantir a acurácia para todas as buscas, requisitando uma memória proporcional a $\frac{1}{1-c}$ (GIONIS *et al.*, 1999).

4.1.2 LSH forest

Em resposta a esses problemas, Bawa *et al.* (2005) propuseram uma adaptação no LSH original, chamado LSH Forest. Nesse novo esquema, as assinaturas têm comprimento máximo k, podendo ter tamanho menor. Cada assinatura é tão longa quanto necessário para que sejam únicas (respeitando o tamanho máximo k). Além disso, a informação de cada uma das l tabelas é guardada de maneira eficiente, em forma de árvore de prefixos (por isso o nome *Forest*). Deste modo, elimina-se a necessidade de ajustar o parâmetro k.

As árvores de prefixos permitem uma compressão da estrutura de indexação, em comparação com simples tabelas de *hashing*. O algoritmo começa por gerar assinaturas de tamanho máximo k para todos os pontos da base (Algoritmo 3, linhas 3 a 6). Em seguida é construída uma árvore de prefixos contendo essas assinaturas (Algoritmo 3, linha 7), levando à compressão do índice. Cada nó folha representa um ponto do conjunto de dados, e o caminho da raiz até ele constitui sua assinatura. A Figura 7 exemplifica uma árvore de prefixos correspondente a uma tabela de *hashings* binários. Se um ponto já pode ser diferenciado dos demais com menos de k bits, então o caminho até ele na árvore — e sua assinatura — serão mais curtos. Isso elimina a necessidade do ajuste fino do tamanho da assinatura. A geração dos índices é repetida l vezes, onde l é um parâmetro de entrada (Algoritmo 3, linha 2).

A busca de vizinhos se dá em duas fases (Algoritmo 4). Primeiro, as árvores de

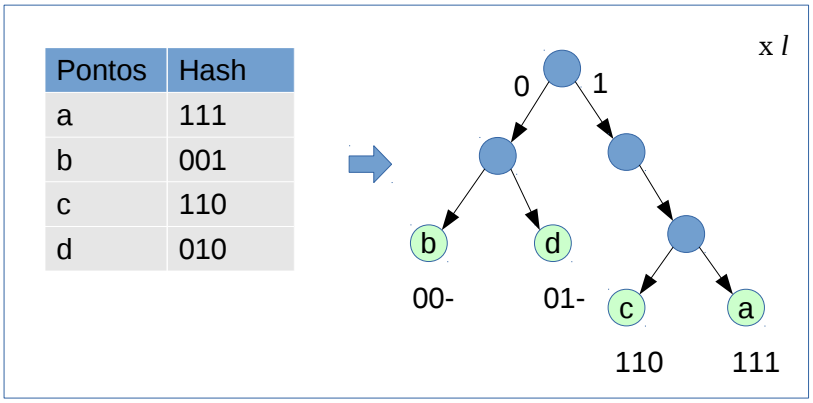


Figura 7 – Tabela de hashing e árvore de prefixos correspondente.

Algoritmo 3: Geração do índice do LSH Forest

Entrada: $X = x^{(1)}, \cdots, x^{(n)}$: dados de entrada, l: num. de árvores
Saída: $T = (T_1, \cdots, T_l)$: conjunto de l árvores LSH

```
1  início
2      para j = 1, ··· , l faça
3          para cada x^(i) ∈ X faça
4              Gerar uma assinatura s_j^(i) de tamanho k
5              S_j ← s_j^(i)
6          fim
7          Construir uma árvore de prefixos T_j para assinaturas em S_j
8          T ← T_j
9      fim
10     retorna T
11 fim
```

prefixos são percorridas buscando-se o maior caminho possível que corresponda à assinatura do ponto de consulta (linhas 3 a 7). Em seguida, a partir do nó designado por esse caminho, percorre-se a árvore de maneira ascendente, recuperando candidatos a vizinho que também correspondem à assinatura, mas por um trecho menor (linhas 8 a 13). Sobe-se um nível por vez, verificando todos os nós folha daquele nível. Por exemplo, na Figura 7, uma consulta pelo ponto 111 vai encontrar a profundidade máxima de *match* no nó (a), nível 3. Nesse nível recupera-se também o nó (c) como candidato. Em seguida sobe-se para o nível 2 e busca-se mais candidatos. Nesse exemplo não há nenhum nó folha coincidindo o prefixo, então a busca ascendente chega ao fim. Se houvesse o ponto 101 por exemplo, ele seria um candidato recuperado no nível 2. Antes de subir, o nível atual é checado em todas as árvores. Para cada árvore são recuperados m candidatos, totalizando ml candidatos no total. Desse modo, m e l são os parâmetros do algoritmo. Por fim é calculada a distância entre o ponto de consulta e os candidatos, retornando os k mais próximos (linhas 14 a 19).

Algoritmo 4: Busca no índice do LSH Forest

Entrada: q: ponto de consulta, T: índice LSH Forest, c: num. candidatos por árvore, k: num. vizinhos, *minmatch*: tamanho mínimo do *match* de assinaturas
Saída: $V = (v_1, \cdots, v_k)$: conjunto de k vizinhos de q

início
  Calcular assinatura s para a consulta q
  para cada *árvore* T_j *em* T, $j = 1, \cdots, l$ **faça**
    $n_j \leftarrow$ nó de máximo *match* de prefixo com s
    $d_j \leftarrow$ profundidade de n_j
  fim
  $maxD \leftarrow \max_j d_j$ `/* máxima profundidade de match */`
  $candidatos \leftarrow$ **enquanto** $maxD > minmatch$ **e** $|candidatos| < cl$ **faça**
    para cada *árvore* T_j *em* T, $j = 1, \cdots, l$ **faça**
      $candidatos \leftarrow$ nós folhas com *match* de prefixo à profundidade $maxD$
    fim
    $maxD \leftarrow maxD - 1$
  fim
  para cada $c \in candidatos$ **faça**
    Calcula similaridade entre q e c
  fim
  Ordena $candidatos$ por similaridade
  $V \leftarrow$ top k $candidatos$
  retorna V
fim

4.2 Construção de grafo kNN aproximado

Chen *et al.* (2009) propuseram um algoritmo estilo divisão-e-conquista. Os dados são divididos recursivamente em subconjuntos com sobreposição. A divisão se dá segundo um hiperplano definido pelo primeiro vetor singular (com maior valor singular), obtido via bisseção de Lanczos (1950). Para cada subconjunto, um grafo kNN é construído. O grafo final é obtido pela conjunção dos grafos menores, baseando-se nas porções sobrepostas. O método apresentou complexidade empírica de $O(Dn^{1,22})$ e se aplica apenas à métrica euclidiana.

Dong *et al.* (2011) propuseram um método de construção rápido baseado em busca local chamado NN-descent. Sua motivação se baseia na heurística de que o vizinho do vizinho é provavelmente também um vizinho. A vizinhança de cada ponto é inicializada aleatoriamente, sendo iterativamente melhorada através da exploração das vizinhanças de seus vizinhos. Foi reportado um custo computacional empírico de $O(n^{1,4})$, mas sem garantias formais da complexidade. Seu método foi comparado com o método de Chen *et al.* (2009) e também com a construção do grafo baseada em busca LSH, apresentando melhores resultados tanto em termos de acurácia quanto de tempo de execução. Contudo, esse bom comportamento não é mantido quando o número de dimensões cresce para além de algumas dezenas.

O LSH também foi aplicado para o problema da construção de grafo kNN (ZHANG

et al., 2013), também dividindo os dados em pequenos grupos usando LSH, construindo um grafo kNN para cada um dos grupos e agrupando tudo em um grafo final. Entretanto, como o LSH geralmente divide os dados em grupos de maneira desigual, alguns grupos podem ficar muito grandes e custosos para a construção do grafo via força bruta. Para equilibrar esse custo, os valores de *hash* de cada ponto são projetados em um hiperplano aleatório, e os pontos são ordenados segundo o valor dessas projeções, para em seguida serem divididos em grupos de tamanhos iguais. Esse método pode ser aplicado a qualquer medida de distância que tenha uma família de funções LSH correspondente.

Park *et al.* (2014) propuseram um método baseado na geração de prefixos que correspondem às maiores dimensões de cada vetor. São considerados candidatos a vizinhos apenas aqueles que têm uma dimensão em comum em seus prefixos. Quando cada vetor tem aproximadamente o mesmo número de vizinhos candidatos (ou se todas as dimensões forem consideradas), o algoritmo procede a construção do grafo via força-bruta ou por um método baseado em um índice invertido. Esse método possui complexidade $O(n)$, e se aplica à similaridade do cosseno. Algumas exigências quanto a distribuição dos dados são impostas, mas segundo os autores, a grande maioria das bases segue essas restrições se for aplicado algum esquema de pesos como TF-IDF (LESKOVEC *et al.*, 2014, pp. 8-9).

Nesse trabalho, foi escolhido o LSH Forest como método de vizinhança aproximada, que traz algumas melhorias em relação o método original, além de estar disponibilizado na biblioteca Python Scikit-learn (PEDREGOSA *et al.*, 2011). Seu uso será descrito na proposta apresentada no capítulo 7.

5 Projeções aleatórias

5.1 Redução de dimensionalidade com projeções aleatórias

A redução de dimensionalidade é uma etapa comumente necessária em diversas aplicações de mineração de dados. Um dos métodos mais utilizados é a projeção dos dados em um sub-espaço ortogonal de menor dimensão. A maneira mais indicada de realizar essa tarefa (em termos de erro quadrático médio) é a análise de componentes principais (*Principal Component Analisys* - PCA). Entretanto, para dados com alta dimensão, este torna-se um procedimento custoso, de modo que um método de menor complexidade algorítmica é desejável, motivando o uso do método de projeções aleatórias (*Random Projections* - RP) (BINGHAM; MANNILA, 2001).

Nessa técnica, dados com alta dimensão (d) são projetados em um subespaço de menor dimensão ($k \ll d$) através de uma matriz aleatória $R \in \mathbb{R}^{d \times k}$. Considerando uma matriz X com m amostras de dados de dimensão d, a projeção se dá na seguinte forma:

$$X_{m \times k} = X_{m \times d} R_{d \times k}, \quad k \ll \min(m, d). \tag{5.1}$$

Em geral, $r_{ij} \sim N(0,1)$.

Tomando dois vetores $x_1, x_2 \in \mathbb{R}^d$, com a distância euclidiana denotada por $\|x_1 - x_2\|$, após a projeção aleatória, a distância entre os pontos é aproximadamente a distância euclidiana no espaço reduzido, multiplicada por um fator:

$$\|x_1 - x_2\| \approx \sqrt{d/k}\,\|Rx_1 - Rx_2\|, \tag{5.2}$$

onde d é a dimensão original e k a dimensão reduzida (BINGHAM; MANNILA, 2001).

Foi mostrado que essa técnica leva a resultados comparáveis àqueles obtidos com outras técnicas de redução de dimensão, trazendo ainda ganhos em complexidade computacional. De fato, formar a matriz aleatória R e projetar a matriz $X_{m \times d}$ em k dimensões tem complexidade $O(mdk)$.

Mais precisamente, essa operação não é uma projeção pois R, em geral, não é ortogonal. Se esse for o caso, uma transformação linear como essa pode inserir distorções significativas nos dados. Entretanto, num espaço de grande dimensão , existe uma proporção maior de direções quase-ortogonais do que não-ortogonais (HECHT-NIELSEN, 1994). Portanto, vetores com direções aleatórias podem ser suficientemente próximos de ortogonais, dispensando a necessidade de ortogonalização da matriz R (BINGHAM; MANNILA, 2001).

5.1.1 Teorema de Johnson-Lindenstrauss

Essa técnica se inspira no teorema de Johnson-Lindenstrauss (JOHNSON *et al.*, 1986), que diz que se vetores de um espaço vetorial são projetados em um subespaço aleatório de dimensão suficientemente alta, as distâncias euclideanas par-a-par entre os pontos são aproximadamente preservadas. Mais precisamente, ele determina a existência de um mapeamento $f : \mathbb{R}^d \to \mathbb{R}^k$ que, para um conjunto $\mathcal{V}$ de n pontos em $\mathbb{R}^d$, mantém as distâncias projetadas numa margem $(1 \pm \varepsilon)$ da distância original, para $0 < \varepsilon < 1$ e $k > k_0 = O(\log(n)/\varepsilon^2))$, com probabilidade p (DASGUPTA; GUPTA, 2002).

Diversas provas desse teorema foram feitas, levando a algumas diferentes expressões para o limite k_0. Essencialmente, parte-se do fato de que, quando um vetor unitário é projetado em um subespaço aleatório k-dimensional, sua norma L2 se concentra em torno de sua média que é $\sqrt{d/k}$. Uma prova detalhada é dada por Dasgupta e Gupta (2002). O melhor limite foi dado por Achlioptas (2003), que prova o teorema para $k > k_0 = (4+2\gamma)(\varepsilon^2/2 - \varepsilon^3/3)^{-1}\log(n)$, com probabilidade $p = 1 - n^{-\gamma}$.

5.1.2 Projeções aleatórias esparsas

Os elementos r_{ij} da matriz R devem ser i.i.d. e com média zero, sendo essa a única condição necessária para a preservação das distâncias par a par (LI *et al.*, 2006). No geral, é conveniente a escolha de uma distribuição simétrica em torno de zero com variância unitária. Uma escolha óbvia é a distribuição normal, uma função simples do ponto de vista teórico. No entanto, do ponto de vista prático, a geração de números aleatórios seguindo essa distribuição é complicada. Além disso, se a matriz R puder ser mais esparsa, teríamos uma redução na complexidade do cálculo da projeção. Achlioptas (2003) propôs o uso de uma matriz de projeção R esparsa, com entradas da seguinte forma:

$$r_{ij} = \sqrt{s} \cdot \begin{cases} +1 & \text{com probabilidade } \frac{1}{2s} \\ 0 & \text{com probabilidade } 1 - \frac{1}{s} \\ -1 & \text{com probabilidade } \frac{1}{2s} \end{cases}, \tag{5.3}$$

onde foi usado $s = 1$ e $s = 3$. Note que a projeção esparsa faz uma amostragem dos dados com uma taxa $\frac{1}{s}$. Com $s = 3$, pode-se atingir uma melhora de velocidade de até 3 vezes, pois apenas $\frac{1}{3}$ dos dados necessita processamento. O método de projeções aleatórias esparsas se tornou popular e foi testado experimentalmente pela primeira vez por Bingham e Mannila (2001) .

Posteriormente, Li *et al.* (2006) propuseram o uso de $s \gg 3$, o que aumenta ainda mais a velocidade de cálculo das projeções. Essa proposta se baseia no fato de que, se os dados forem aproximadamente gaussianos, uma taxa de amostragem $s = \frac{d}{\log d}$ é provavelmente suficiente. No entanto, os autores recomendam escolher uma taxa menos extrema, de $s = \sqrt{d}$ por exemplo. Foi demonstrado que assintoticamente esse tipo de projeção super-esparsa apresenta

a mesma performance que uma projeção com distribuição gaussiana. Mesmo com dados de distribuição com cauda pesada (*heavy-tailed*), a performance é razoável, mas fica mais próxima da performance da projeção aleatória tradicional se for aplicado algum esquema de pesos nos dados, como TF-IDF (LESKOVEC *et al.*, 2014, pp. 8-9), para deixar a distribuição mais uniforme.

5.2 Projeções aleatórias em ensembles

O uso de múltiplas projeções aleatórias é uma técnica que já foi utilizada na literatura em diferentes contextos. Um dos usos é a própria busca de vizinhos aproximada (discutida no capítulo 4), já que a projeção em um hiperplano aleatório consiste em uma família de funções LSH para a distância do cosseno. Para aumentar a robustez da busca, múltiplas projeções são realizadas.

Zhao e Mao (2015) propõem um método de redução de dimensão semi-aleatório, baseado em múltiplas projeções aleatórias em um subespaço de menor dimensão. Sobre o resultado de cada uma, um método de redução de dimensão determinístico – como LDA ou PCA — é utilizado para extrair uma representação unidimensional dos dados. O conjunto de vetores obtidos formará a nova representação reduzida dos dados.

Fern e Brodley (2003) e Fern *et al.* (2006) introduziram o uso de múltiplas projeções aleatórias na geração de um ensemble de clusters. Nesses trabalhos, os dados são projetados em diferentes subespaços de mesma dimensão, seguindo-se múltiplas clusterizações via *k-means*. Bertoni e Valentini (2006) propõem um método similar, utilizando a clusterização hierárquica como algoritmo-base (WARD, 1963). Avogadri e Valentini (2009) usam múltiplas projeções aleatórias antes de realizar uma clusterização *fuzzy*, no contexto de análise de dados de expressão gênica.

Múltiplas projeções aleatórias também foram utilizadas para gerar ensembles de SVMs, treinados a partir de diferentes projeções dos dados (MAUDES *et al.*, 2010; MAUDES *et al.*, 2011). Schclar e Rokach (2009) também utilizam RP para reduzir a dimensão dos dados de entrada antes de treinar múltiplos classificadores kNN, produzindo resultados competitivos.

Nesse trabalho, múltiplas projeções aleatórias também serão utilizadas para gerar componentes de um *ensemble*. As componentes serão geradas a partir de matrizes obtidas por fatoração booleana, e o *ensemble* visará a predição de notas no contexto de sistemas de recomendação. Tanto a projeção gaussiana quanto a esparsa serão testadas. Essa linha de trabalho será detalhada no capítulo 7.

6 Redes complexas em análise de atividade cerebral

A análise do cérebro com base em modelos de rede complexa é um paradigma recorrente na literatura. É sabido que os neurônios formam uma intrincada e complexa rede estrutural, e também foi observada a emergência de comportamentos coerentes entre regiões espacialmente distantes, indicando a existência de redes funcionais. Tudo isso tem motivado o paradigma de análise do cérebro como uma rede complexa, uma representação em rede com propriedades topológicas particulares como mundo pequeno e lei de potência (BULLMORE; SPORNS, 2009).

A natureza dos nós e arestas em uma rede cerebral varia segundo os métodos de mapeamento cerebral utilizados. Em geral, diferentes nós representam regiões cerebrais distintas, e cada nó agrupa áreas do cérebro anatômica ou funcionalmente correlatas. Existem múltiplas abordagens de segmentação adotadas, que levam a redes com diferentes propriedades topológicas. As arestas podem denotar uma conexão anatômica ou funcional e, dependendo do contexto, podem ser ponderadas e direcionadas, indicando um grau de correlação ou causalidade entre as regiões conectadas (RUBINOV; SPORNS, 2010).

Nas últimas décadas, métodos que analisam a conectividade funcional têm sido cada vez mais usados, tanto na caracterização da organização normal de um cérebro sadio, quanto na identificação de alterações devidas a diversos transtornos neuronais (FALLANI *et al.*, 2014). Em geral, a atividade de diferentes regiões é medida através de alguma técnica de registro de atividade cerebrais, como a ressonância magnética funcional (*functional magnetic resonance imaging* – fMRI), a eletroencefalografia (*electroencephalography* – EEG) e a magnetoencefalografia (magnetoencephalography – MEG). A magnitude da dependência temporal entre essas regiões descreve como elas interagem funcionalmente. Considerando N regiões, teremos $N \times N$ relações funcionais que levam à matriz de adjacências que define a rede.

A dependência temporal entre as séries de diferentes regiões pode ser medida de diversas formas, como pela correlação de Pearson ou pela informação mútua. Essas medidas podem determinar o peso das arestas e, em geral, arestas com peso muito baixo — abaixo de um limiar t — são desconsideradas. Pode-se também utilizar o limiar t para binarizar a matriz de adjacências, gerando redes com arestas sem pesos. Não há um método sistemático para a escolha desse limiar, por isso normalmente as redes são geradas e analisadas para vários valores de t (FALLANI *et al.*, 2014). A Figura 8 apresenta um fluxograma geral das abordagens de análise em redes.

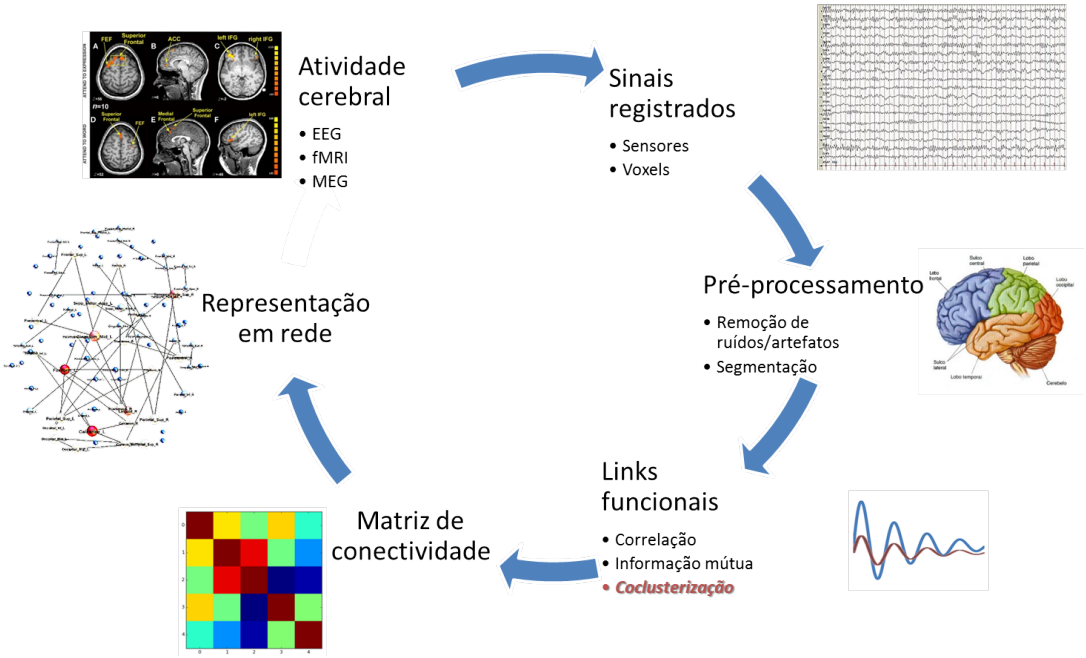


Figura 8 – Fluxograma geral das abordagens de análise do cérebro via redes complexas, com a co-closterização inserida como ferramenta para a determinação de links funcionais (inspirado em Fallani *et al.* (2014))

6.1 Métricas

Diversas métricas têm sido utilizadas no contexto de redes cerebrais, como forma de caracterizar diferentes estados do cérebro. Existem tanto métricas globais, que resumem características gerais da rede, como métricas locais, que informam sobre a participação de nós ou pequenos grupos. Rubinov e Sporns (2010) apresentam uma revisão detalhada das principais métricas de rede utilizadas nesse cenário. A seguir são descritas as métricas utilizadas neste trabalho. Nas definições seguintes, N é o conjunto de nós do grafo e $n = |N|$ é o número de nós do grafo.

6.1.1 Densidade

A densidade de rede é o grau médio de todos os nós da rede, e reflete seu nível de conectividade. É definida por:

$$D = \frac{1}{n} \sum_{i \in N} e_i, \tag{6.1}$$

onde e_i é o grau do nó i, ou seja, o número de arestas ligadas a ele.

6.1.2 Coeficiente de clusterização e clusterização média

O coeficiente de clusterização é uma métrica local, em nível de um nó individual, e consiste na fração de triângulos no seu entorno. O coeficiente de clusterização para um nó i é definido da seguinte forma:

$$C(i) = \frac{2t_i}{e_i(e_i - 1)}, \tag{6.2}$$

onde t_i é o número de triângulos no entorno do nó i e e_i é o grau do nó i.

A clusterização média é definida como a média do coeficiente de clusterização para todos os nós da rede:

$$\bar{C} = \frac{1}{n}\sum_{i \in N} C(i). \tag{6.3}$$

Uma alta clusterização média reflete a prevalência, na média, de conexões aglomeradas em grupos de nós. É considerada uma medida de segregação funcional: a ocorrência de processamento interno a grupos de nós densamente interconectados.

6.1.3 Comprimento característico e eficiência global

A média do comprimento do caminho mais curto entre todos os pares de nós de uma rede é conhecido como comprimento característico da rede.

$$L = \frac{\sum_{j \in N, j \neq i} d_{ij}}{n-1} \tag{6.4}$$

Uma outra medida relacionada é a eficiência global que é a média dos inversos dos comprimentos. Ela tem a vantagem de poder ser calculada em redes desconectadas, enquanto o comprimento característico não admite esse tipo de rede: a distância entre nós de componentes desconexos é infinita mas seu inverso é zero. A eficiência global é definida da seguinte forma:

$$E = \frac{1}{n}\sum_{i \in N} E_i = \frac{1}{n}\sum_{i \in N} \frac{\sum_{j \in N, j \neq i} d_{ij}^{-1}}{n-1}, \tag{6.5}$$

onde E_i é a eficiência do nó i, e d_{ij} a menor distância entre os nós i e j.

Os caminhos em uma rede representam rotas de fluxo de informação entre regiões. Por isso, essas são ditas medidas de integração funcional e refletem a capacidade do cérebro de combinar rapidamente informação de regiões distribuídas. Quanto mais curtos os caminhos, maior o potencial de integração de informação da rede e maior sua eficiência global.

6.1.4 Centralidade

Medidas de centralidade indicam a importância de nós individuais na rede. Nós tidos como importantes em geral são aqueles que interagem com muitos outros nós, facilitando a integração de informação e/ou trazendo robustez a perturbações ou falhas da rede.

Existem diversas medidas de centralidade, e a centralidade de grau é a mais comum delas. Ela é a proporção de nós da rede conectados ao nó dado. Quanto maior o grau, mais o nó interage com outros nós da rede. Matematicamente:

$$C_d(i) = \frac{e_i}{n_A - 1}, \tag{6.6}$$

onde e_i é o número de arestas ligadas ao nó i e n_A o número de arestas no grafo.

Existem também medidas de centralidade que consideram a participação de um nó nos caminhos mais curtos da rede. Quanto maior essa participação, mais esses nós estão intermediando os principais fluxos de informação da rede. Nesse intento, temos a centralidade de proximidade, definida como o inverso da média dos caminhos mais curtos entre um dado nó e todos os demais:

$$C_c(i) = \frac{n_A - 1}{\sum_{j \in N, j \neq i} d_{ji}}, \tag{6.7}$$

onde d_{ij} é a menor distância entre os nós i e j. Um nó que esta a uma curta distância de muitos nós tende a ter uma centralidade de proximidade mais alta.

Outra medida de centralidade nesse sentido é a centralidade de *betweeness*, definida como a fração dos caminhos mais curtos da rede que passam por um dado caminho:

$$C_b(i) = \frac{1}{(n_A - 1)(n_A - 2)} \sum_{h,j \in N, i \neq j \neq h} \frac{\rho_{hj}(i)}{\rho_{hj}}, \tag{6.8}$$

onde ρ_{hj} é o número de caminhos mais curtos entre h e j, e $\rho_{hj}(i)$ é a quantidade desses caminhos passando por i. Essa centralidade indica que um nó age como *hub*, concentrando grande parte do fluxo de informação da rede, já que os principais caminhos passam por ele.

6.2 Conectividade funcional dinâmica

Análises estáticas da conectividade funcional são bastante comuns e podem prover informações importantes quanto ao funcionamento normal ou anormal do cérebro. Contudo, cada vez mais há um interesse em estudar as variações de conectividade através do tempo, a fim de ampliar a compreensão dos padrões de atividade cerebral. Hutchison *et al.* (2013) apresentam uma extensa revisão de esforços em análise de conectividade funcional dinâmica. Alterações na dinâmica da conectividade funcional foram reportadas em diversos contextos, como realização de tarefas, aprendizado, sono, anestesia e patologias.

Mais recentemente, considerou-se que as análises de conectividade funcional realizadas tinham uma limitação implícita de assumir estacionariedade temporal e espacial (HUTCHISON *et al.*, 2013). Nesse contexto, Calhoun *et al.* (2014) definem o termo "chronnectome" (cronectoma, em tradução livre) para se referir a modelos que consideram variações temporais nos nós e conexões da rede, supondo assim a não-estacionariedade da dinâmica do sistema. Modelos nesse paradigma têm trazido resultados interessantes e são mais naturalmente adaptados a análise de dados de fMRI de atividade cerebral, dada a sua característica dinâmica e suscetível a condições internas e externas (CALHOUN; ADALI, 2016; HUTCHISON *et al.*, 2013).

Em um trabalho recente de revisão na área, Calhoun e Adali (2016) organizam a análise de conectividade funcional dinâmica em 4 etapas :

1. Dados de entrada: em fMRI, podem ser séries de voxels, de regiões segmentadas segundo um atlas padrão, ou séries agregadas obtidas por um método de separação cega de fontes, como ICA (HYVÄRINEN; OJA, 2000).
2. Inferência das medidas dinâmicas de conectividade: podem ser calculadas por janelamento ou utilizando alguma técnica de decomposição em tempo e frequência (CALHOUN; ADALI, 2016).
3. Inferir estados característicos a partir das matrizes de conectividade (ex: por clusterização, PCA, ICA ou outras técnicas não-supervisionadas).
4. Caracterizar a atividade dinâmica: analisar características da rede de conexões em cada estado, ou analisar as transições entre estados.

O método mais utilizado para obter medidas de conectividade dinâmicas é o janelamento (HUTCHISON *et al.*, 2013). O nível de relação entre a atividade de duas regiões é medido dentro de uma janela deslizante de comprimento fixo ou adaptativo. É comum o uso de janelas ponderadas, por uma gaussiana por exemplo (CALHOUN; ADALI, 2016; CALHOUN *et al.*, 2014). Trata-se de uma abordagem simples, por isso já começam a ser exploradas outras possibilidades. Por exemplo, Lindquist *et al.* (2014) criticam o uso dessa técnica e propõe o uso de modelos mais complexos, advindos da literatura de finanças.

Definida a conectividade dinâmica, existem múltiplas possibilidades de análise sob o paradigma de redes complexas. Nesse caso, tem-se uma rede de múltiplas camadas, cada camada representando uma janela temporal. Pode-se realizar o cálculo de métricas topológicas para cada camada, obtendo um perfil de sua variação temporal (FALLANI *et al.*, 2014). Por exemplo, Bassett *et al.* (2011) analisam mudanças na modularidade da rede durante o aprendizado. Fallani *et al.* (2008) analisaram diversas métricas como densidade da rede e grau dos nós, identificando diferenças nos padrões de atividade durante um experimento motor. Chiang *et al.* (2016) analisam a estacionariedade de métricas topológicas, mostrando que algumas são mais estacionárias que outras ao longo do tempo, sendo portanto mais robustas.

A abordagem de análise de do cérebro como uma rede complexa será utilizada nesse trabalho, no capítulo 8. A coclusterização contígua será empregada como método de identificação de regiões co-funcionais, e também de determinação automática da janela de tempo na qual essa co-funcionalidade está presente. Diferentes métricas topológicas serão utilizadas a fim de verificar o potencial das redes de diferenciar indivíduos controle de pacientes. Essa frente de trabalho será detalhada no capítulo 8.

Parte II

Propostas

7 Comitês de filtros colaborativos baseados em BMF e RP

Modelos de fatoração de matriz têm sido usados com sucesso na síntese de filtros colaborativos, mas a fatoração booleana foi pouco explorada nesse contexto. Para bases binárias, esse método é particularmente interessante por gerar tanto a fatoração quanto a reconstrução binárias, o que não ocorre com a decomposição SVD, por exemplo.

Nesse trabalho, nos inspiramos na proposta de Ignatov *et al.* (2014), em que pela primeira vez a fatoração de matriz booleana baseada em FCA foi aplicada em sistemas de recomendação. Propomos um *ensemble* baseado em múltiplas projeções aleatórias da matriz de fatores obtida com a fatoração booleana, a fim de produzir recomendações mais acuradas e robustas (DRUMOND; VON ZUBEN, 2016).

A projeção em um subespaço selecionado ao acaso é uma técnica eficiente de redução de dimensionalidade, que tende a preservar distâncias par-a-par definidas em espaços de elevada dimensão. Contudo, foi mostrado que projeções aleatórias podem levar a resultados bastante distintos quando aplicadas em uma tarefa de clusterização (FERN *et al.*, 2006), motivando a criação de um *ensemble*. Também foi estudada a influência do uso de um método de vizinhança aproximada, a fim de verificar seus efeitos na performance e no tempo de treinamento.

Esse capítulo descreve os estudos realizados e resultados obtidos nessa frente de trabalho. A seção 7.1 detalha a metodologia adotada, seguida pela seção 7.2, que apresenta os experimentos realizados e analisa os resultados. Considerações finais referentes a essa linha de trabalho são apresentadas na seção 7.3.

7.1 Metodologia

Primeiramente, foram realizados experimentos a fim de testar a proposta de sistema de recomendação baseado em fatoração booleana (BMF) de Ignatov *et al.* (2014) (ver seção 3.2.2.2). Para isso, foi implementada a segunda proposta de algoritmo de Belohlavek e Vychodil (2010) (ver seção 2.3), que busca os fatores (coclusters) de maneira gulosa a fim de definir a fatoração booleana. Também foi implementada a recomendação baseada em BMF e seu ambiente de testes.

A recomendação é realizada conforme descrito na seção 3.2.2.2, utilizando um esquema baseado em vizinhança. A matriz de avaliações M é binarizada e em seguida decomposta em duas matrizes P e Q através do método de BMF apresentado na seção 2.3. Com as linhas da matriz P sendo os vetores de cada usuário, essa matriz é utilizada para o cálculo de vizinhanças

entre usuários. Dessa forma, para um dado usuário alvo, é utilizado seu vetor correspondente na matriz P para buscar top-k vizinhos que tenham avaliado o item alvo (que se deseja recomendar). Por fim, retorna-se à matriz de avaliações original M e recuperam-se as avaliações do item alvo feitas pelos top-k vizinhos. A predição da avaliação do usuário alvo é feita pela média dessas avaliações dos vizinhos, ponderada pela similaridade calculada com base na matriz P.

A busca de coclusters para a formação das matrizes de fatores pode ser parada prematuramente, segundo um parâmetro de cobertura mínima m: se já foram encontrados fatores suficientes para cobrir uma fração m dos "1"s da matriz original, a busca é interrompida.

A fim de acelerar a seleção de vizinhos na etapa de recomendação, na etapa de treinamento foi gerado um grafo kNN com $l >> k$ vizinhos. Como serão necessários k vizinhos que co-avaliaram o item-alvo para predizer sua nota, é necessário gerar um grafo com mais de k vizinhos, aumentando a chance de ter pelo menos k co-avaliadores. A razão $l/|U|$ define o tamanho do modelo kNN (onde $|U|$ é a quantidade de usuários da base), e quando $l/|U| = 1$ esse método é equivalente a realizar a busca *online*. Ainda assim, a geração prévia do grafo adianta o cálculo das distâncias e busca dos k vizinhos, reduzindo o tempo de recomendação.

O experimento foi repetido utilizando-se busca kNN aproximada, baseada em LSH. Como método de vizinhança aproximada, foi utilizada a implementação da biblioteca Python Scikit-learn (PEDREGOSA *et al.*, 2011), baseada em Bawa *et al.* (2005). Um outro caminho seguido a partir da proposta de Ignatov *et al.* (2014) foi aplicar a redução de dimensão via RP sobre a matriz obtida pela BMF. Deste modo, reduz-se o peso computacional da busca de vizinhos. Essa metodologia foi testada tanto com o método com vizinhança exata quanto com o método aproximado.

Inicialmente, segue-se a metodologia anterior. Entretanto, antes do uso da matriz P para a busca de vizinhos, sua dimensão é reduzida através de projeções aleatórias. A matriz P é multiplicada por uma matriz aleatória R, que leva à redução de dimensão. Dois métodos foram testados: com uma matriz de distribuição gaussiana, e com uma matriz esparsa segundo proposta de Li *et al.* (2006) (com $s = 1/\sqrt{\text{num. dimensões}}$, ver seção 5.1.2).

Como os resultados da projeção podem ser bem diferentes, e como seu cálculo é barato computacionalmente, também propusemos a geração de um *ensemble* a partir de diferentes projeções (Figura 9). Foi realizada uma combinação de notas através de uma regressão com regularização *elastic net* (ZOU; HASTIE, 2005), da seguinte forma:

$$\min_w \frac{1}{2N}||y - Xw||_2^2 + \lambda\alpha||w||_1 + \lambda(1-\alpha)||w||_2^2 \tag{7.1}$$

com $\alpha \in (0,1)$, $\lambda \in \mathbb{R}$. X é uma matriz $N \times c$ e y um vetor $N \times 1$, onde N é o número de triplas usuário-item-nota do conjunto de ajuste do ensemble, e c o número de preditores sendo agregados no *ensemble*. X contém as predições de nota realizadas para cada par (u,i), e y a nota correta.

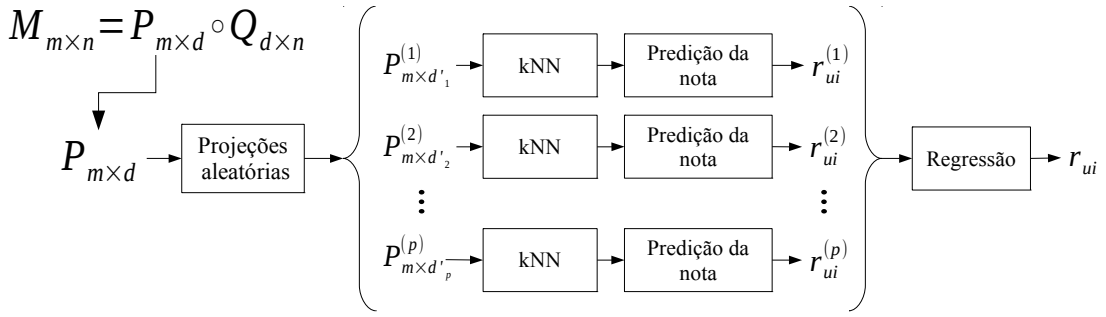


Figura 9 – Diagrama dos módulos constituintes da metodologia de *ensemble*.

A Figura 9 sumariza o *framework* de *ensemble* proposto. Primeiro, a matriz de avaliações binarizada é decomposta em duas matrizes P e Q. Em seguida, a matriz P é projetada em subespaços reduzidos de diferentes dimensões. Cada nova matriz é usada para busca de vizinhos e predição das notas faltantes da base, que são agregadas pela regressão *elastic net*.

7.1.1 Dados

7.1.1.1 Base MovieLens100k

Foi utilizada a base Movielens 10k[1], com 100 mil avaliações de 943 usuários para 1682 filmes (notas de 1 a 5). Nessa versão da base, cada usuário avaliou pelo menos 20 itens. Como a base não é binária, é preciso escolher um limiar para binarizar, e todos os valores foram testados (0 a 4).

As 100 mil avaliações foram divididas em 5 pastas de validação cruzada (divisão por usuário). Das 4 pastas utilizadas para o treinamento, 10% foi reservado para o ajuste do *ensemble*: seleção de componentes e treinamento da regressão *elastic net*.

7.1.1.2 Base Delicious2k

Também foi utilizada a base binária Delicious[2], com 105000 *bookmarks* de 1867 usuários, para mais de 30 mil URLs principais. Devido ao grande número de itens, a validação seguiu um protocolo mais simples: 20% dos dados foram separados para o teste. Do restante, 20% ficam para validação e o restante para treinamento. Também foram separados 10% dos dados de treinamento para ajuste dos pesos do *ensemble*

7.1.2 Avaliação dos resultados

Para avaliar o desempenho das propostas, foram utilizadas métricas de acurácia, cobertura e tempo de treinamento e recomendação, todas tradicionalmente usadas em sistemas de recomendação (SHANI; GUNAWARDANA, 2011). Como métrica de acurácia, foram utilizadas

[1] Disponível em : <http://grouplens.org/datasets/movielens/>
[2] <http://grouplens.org/datasets/hetrec-2011/>

métricas de erro de predição de notas, RMSE e MAE, definidos à seguir:

$$RMSE = \sqrt{\frac{1}{N}\sum_{r_{ui}\in T}^{N} |r_{ui} - \hat{r}_{ui}|^2}, \tag{7.2}$$

$$MAE = \frac{1}{N}\sum_{r_{ui}\in T}^{N} |r_{ui} - \hat{r}_{ui}|, \tag{7.3}$$

onde T é o conjunto de teste e N sua cardinalidade. Historicamente o MAE foi muito utilizado em sistemas de recomendação, mas posteriormente o uso do RMSE se tornou mais comum. O RMSE tem a vantagem de dar um peso relativo maior a erros grandes.

Também foi utilizado o RMSE calculado por nota. Calcula se o RMSE normalmente, mas considerando apenas as notas r_{ui} iguais a nota r que se deseja analisar.

$$RMSE_r = \sqrt{\frac{1}{N}\sum_{r_{ui}\in T | r_{ui}=r}^{N} |r_{ui} - \hat{r}_{ui}|^2} \tag{7.4}$$

Um erro médio mais baixo não necessariamente leva a listas de melhor desempenho. Por isso também foram utilizadas métricas de precisão e revocação (em inglês *recall*), além da métrica F1 que combina as duas — daqui em diante chamadas *métricas de lista*. Tais métricas foram calculadas com base em listas de recomendação de tamanho $N = 20$, da seguinte forma:

$$P@20 = \frac{\textit{num. acertos}}{N}, \tag{7.5}$$

$$R@20 = \frac{\textit{num. acertos}}{\textit{num. itens bons na validação}}, \tag{7.6}$$

$$F1@20 = \frac{2\,P\,R}{P+R}. \tag{7.7}$$

$$\tag{7.8}$$

Foram considerados bons itens com nota maior que 3 e o conjunto de validação contém tanto itens bons quanto ruins.

7.1.3 Comparação

Como parâmetro de comparação de desempenho dos métodos propostos, alguns métodos tradicionais de filtragem colaborativa foram implementados. São os seguintes:

- Método baseado em item (IB), conforme seção 3.2.1. Nesse método foi utilizada a similaridade do cosseno ajustado.
- Método baseado em usuário (UB), conforme seção 3.2.1. Nesse método foi utilizada a similaridade de correlação de Pearson.

- SVD, conforme seção 3.2.2.1. Foi utilizado o método regularizado, conforme equação 3.9, e a predição das notas dada pela equação 3.8. Note que esse modelo não é o mesmo utilizado em Ignatov *et al.* (2014), que utiliza SVD seguido de uma recomendação por vizinhança.

7.2 Resultados e discussão

7.2.1 Experimentos na base MovieLens 100k

7.2.1.1 Análise de parâmetros da recomendação com BMF

Nos experimentos baseados em BMF, alguns parâmetros foram variados na validação cruzada:

- Número de vizinhos k
- Tamanho do modelo kNN $l/|U|$
- Cobertura mínima m
- Limiar de binarização t

A métrica de similaridade foi a do cosseno.

A Figura 10 mostra a performance de diferentes coberturas mínimas, variando o número de vizinhos. Essa figura e as seguintes mostram valores médios sobre as 5 pastas da validação, indicando com barras de erro o desvio padrão. Considerando número de vizinhos, 30 leva ao menor RMSE. O melhor limiar de binarização foi $t = 3$ para todas as métricas. Esses valores foram utilizados em testes subsequentes.

Na análise do tamanho de modelo kNN, sua redução não piora significativamente o RMSE até $l = 0,6|U|$ (Figura 11). Contudo, a qualidade das listas piora quase linearmente. Para eliminar esse efeito no estudo de outros parâmetros, foi mantido $l = |U|$.

Na Tabela 1, temos a quantidade de fatores computados para as diferentes coberturas mínimas. Mesmo com cobertura total, há uma redução expressiva em comparação a um método baseado em usuário comum, cuja dimensão seria o número de itens $|I|$. Contudo, o tempo para obter as matrizes de fatores é alto, e compõe a maior parte do tempo de treinamento. Pode-se obter tempos menores e reduções mais expressivas parando a busca de conceitos prematuramente. Note que uma cobertura de 80% já atinge uma grande redução com uma pequena piora no erro e quase sem afetar a precisão. Nas análises seguintes, mantivemos $m = 100\%$ para eliminar seu efeito sobre os resultados, mas os dados da Tabela 1 indicam que o uso de uma cobertura mais baixa para reduzir o tempo de treinamento pode ser um bom compromisso.

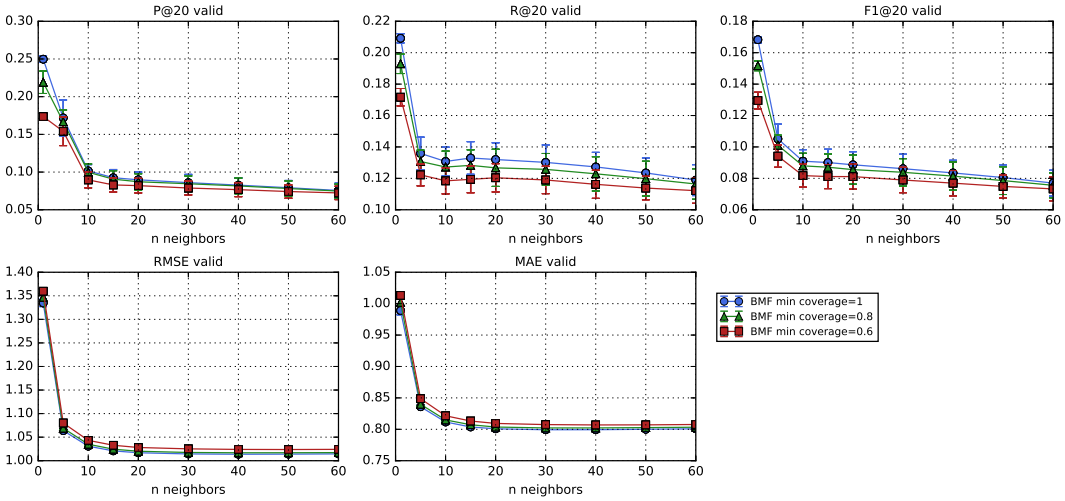


Figura 10 – Performance de diferentes coberturas mínimas, variando o número de vizinhos (validação).

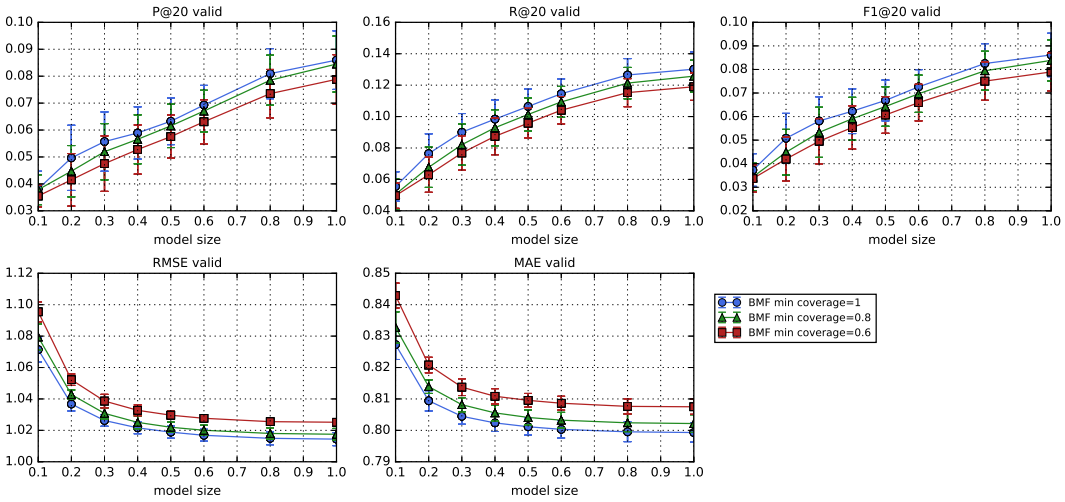


Figura 11 – Performance de diferentes coberturas mínimas, variando o tamanho de modelo kNN (validação).

Tabela 1 – Número de fatores encontrados para diferentes coberturas mínimas (média dos valores obtidos na validação), relacionados com o número de usuários $|U| = 943$ e de itens $|I| = 1682$, e com o tempo gasto para calcular as matrizes.

Cobertura mínima	Número de fatores f	$f/\|U\|$	$f/\|I\|$	Tempo(s)	$P@20$	RMSE
100%	1119 ± 11	1,187	0,666	1738 ± 350	$0{,}09 \pm 0{,}01$	$1{,}014 \pm 0{,}004$
80%	363 ± 3	0,386	0,216	403 ± 5	$0{,}08 \pm 0{,}01$	$1{,}018 \pm 0{,}004$
60%	202 ± 1	0,215	0,120	229 ± 3	$0{,}08 \pm 0{,}01$	$1{,}025 \pm 0{,}002$

7.2.1.2 Análise de parâmetros da recomendação BMF com vizinhança aproximada LSH

No algoritmo LSH Forest, foram considerados dois hiper-parâmetros:

- número de candidatos e
- número de estimadores.

O número de candidatos se refere a quantos candidatos a vizinho serão considerados por estimador, e o número de estimadores é o número de árvores que serão consideradas para a deter-

minação da vizinhança. Os dois parâmetros também foram variados na validação.

O número de candidatos precisa ser pelo menos igual ao número k de vizinhos que se busca. No caso de geração de grafo *offline*, usa-se um número de vizinhos $l \geq k$. Alguns valores nl foram testados, variando também o número de estimadores.

A Figura 12 mostra a variação do número de candidatos, e a Figura 13 mostra a variação do número de estimadores, para diferentes tamanhos de modelo kNN. O número de candidatos tem relativamente pouca influência na performance. Já o número de estimadores pode ter uma influência um pouco maior entre 10 e 20, principalmente para $l = 0,3|U|$ e $l = 0,5|U|$. Seguindo o que foi usado em Bawa *et al.* (2005), ficamos com um número de candidatos igual a $2k$ e número de estimadores igual a 10, que é suficiente se usado o tamanho de modelo $l = |U|$.

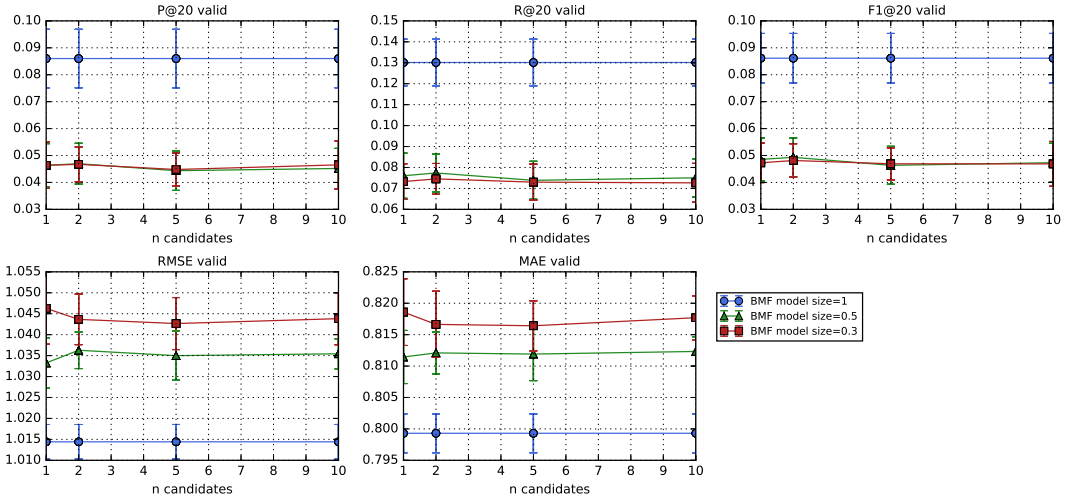


Figura 12 – BMF com LSH, variando número de candidatos e tamanho de modelo kNN. O número de candidatos indicado corresponde apenas ao múltiplo n, de modo que o número de candidatos de fato é nl.

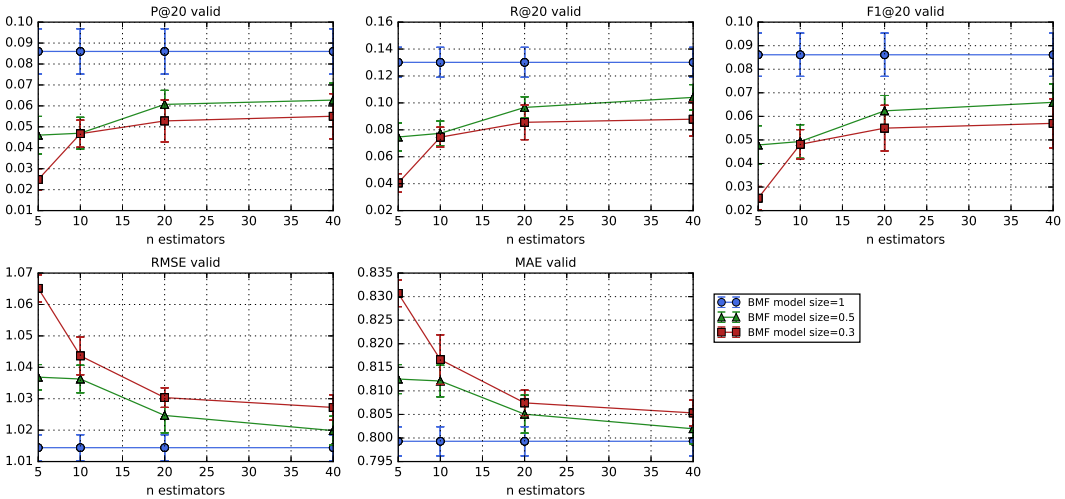


Figura 13 – BMF com LSH, variando número de estimadores e tamanho de modelo kNN

7.2.1.3 Redução de dimensão com RP

Na redução de dimensão com RP, também foram explorados os seguintes parâmetros:

- Projeções aleatórias ou esparsas (esparsas usando $s = \sqrt{d}$, ver seção 5.1.2),
- Nova dimensão reduzida $d' = rd$, com $r \in (0,1)$,
- Uso de ponderação TF-IDF.

O uso de RP teve pouco impacto na acurácia da recomendação (ver Tabela 2). Tanto projeções gaussianas como esparsas tiveram performances similares (ver Figura 14). As métricas de lista apresentam uma grande variância, sendo um bom indicador de que múltiplas RPs podem trazer diversidade. Em termos de erro médio, ele é estável até $d' = 50\%d$, aumentando apenas para reduções mais extremas.

A matriz P obtida por BMF é bastante esparsa (apenas 2% de 1's) e, portanto, com uma distribuição de valores bastante assimétrica. Li *et al.* (2006) sugere que em casos como esse é possível que o uso de uma ponderação TF-IDF melhore os resultados da projeção esparsa. Nos testes realizados, houve uma pequena melhora na média, mas os grandes desvios padrões inviabilizam afirmações conclusivas.

O tempo de treinamento sofre um pequeno aumento em relação ao BMF sozinho (ver Tabela 2). Isso pode ser explicado pelo fato de que a redução de dimensão obtida pode não compensar o gasto computacional para realizar a projeção, especialmente para bases pequenas.

Também foi testado o uso de vizinhança aproximada (LSH) com projeção aleatória esparsa. Há pouca variação no resultado tanto em termos de erro quanto nas métricas de lista, em relação a usar apenas a RP. Além disso, o tempo de recomendação sofreu um pequeno aumento em relação ao uso de RP com vizinhança exata (Tabela 2).

Comparando com os métodos IB e UB, tanto o uso de BMF quanto o de BMF+RP tiveram um desempenho competitivo para vários valores de k (número de vizinhos), como mostrado na Figura 15. Os erros foram ligeiramente maiores, mas nas métricas de lista houve um melhor desempenho.

7.2.1.4 *Ensemble* com *elastic net*

Para a regressão regularizada, foram explorados alguns valores para os seguintes parâmetros:

- coeficiente de regularização λ
- coeficiente de ponderação da *elastic net* α

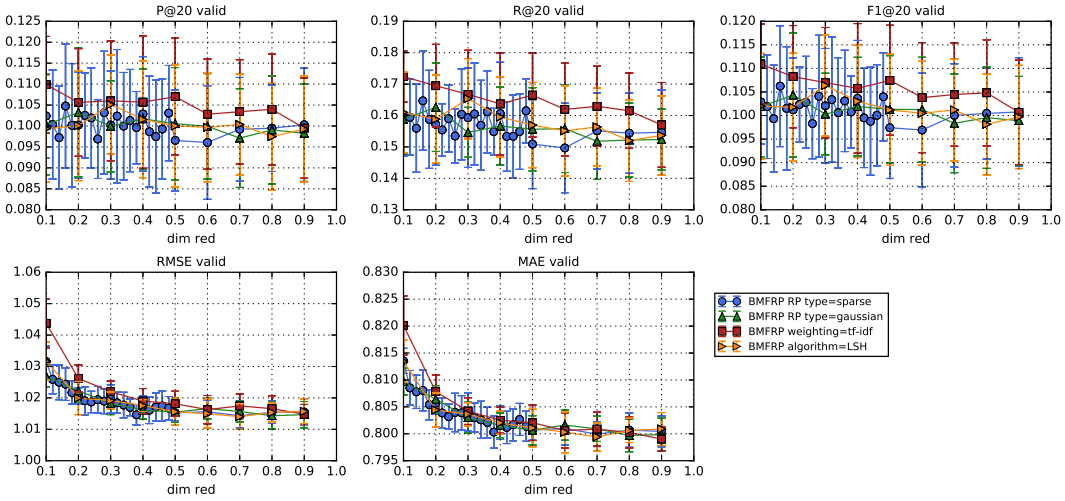


Figura 14 – Métricas de erro e lista para modelo BMF com redução de dimensão RP (validação). Foram usados 30 vizinhos.

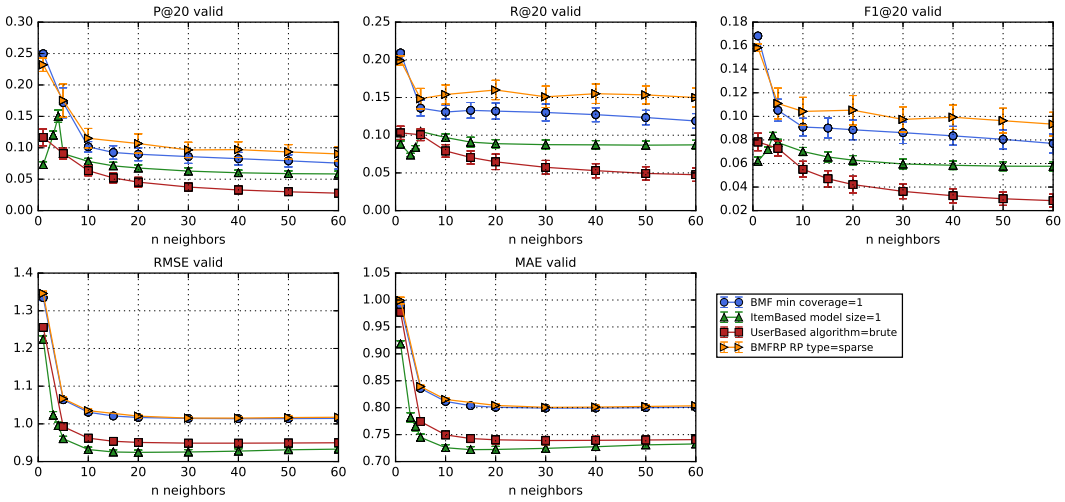


Figura 15 – Comparação da recomendação BMF, BMF+RP(50%), IB e UB, para várias quantidades de vizinhos (validação).

A Figura 16 mostra alguns parâmetros de regularização testados para a *elastic net*. Foram utilizados $\lambda = 0,1$ e $\alpha = 0,5$.

Devido à pequena diferença de performance entre as diferentes reduções (Figura 14), foi testada a geração do *ensemble* unindo 11 componentes BMF+RP com reduções expressivas, mantendo entre 10 e 30 % da dimensão original.

Nas métricas de lista, o desempenho do *ensemble* foi superior aos modelos BMF, BMF+LSH, BMF+RP, BFM+RP+LSH, UB e IB, sendo inferior ao desempenho do SVD (ver Figura 17). O erro médio de predição é relativamente alto. Por outro lado, observando o erro RMSE por nota (Figura 18), vemos que o *ensemble* apresenta um RMSE mais baixo que os outros modelos na nota 3, que é a nota limiar de decisão para um item entrar ou não na lista. Provavelmente isso leva à produção de listas mais precisas.

Pode-se notar uma grande variância na performance, provavelmente devida às projeções aleatórias. Isso combinado à boa performance na nota 3 em detrimento da performance nas notas mais altas pode indicar que o ensemble está regredindo à média, tendendo a imputar mais notas 3.

Os tempos de treinamento para os métodos com BMF são muito superiores aos tempos para os métodos IB,UB ou SVD, devido ao alto custo da realização da BMF (ver Tabela 3). O tempo de treinamento do *ensemble* (regressão) é bastante baixo em relação ao tempo de treinamento dos modelos de base. Sendo possível o treinamento destes em paralelo, o *ensemble* não agrega custos computacionais significativos em relação ao tempo de treinamento de um modelo de base individual. Por isso na Tabela 2 o tempo do *ensemble* é reportado como $\Delta + T$, onde Δ é o tempo de treinamento dos modelos de base e T o tempo de treinamento do *ensemble*.

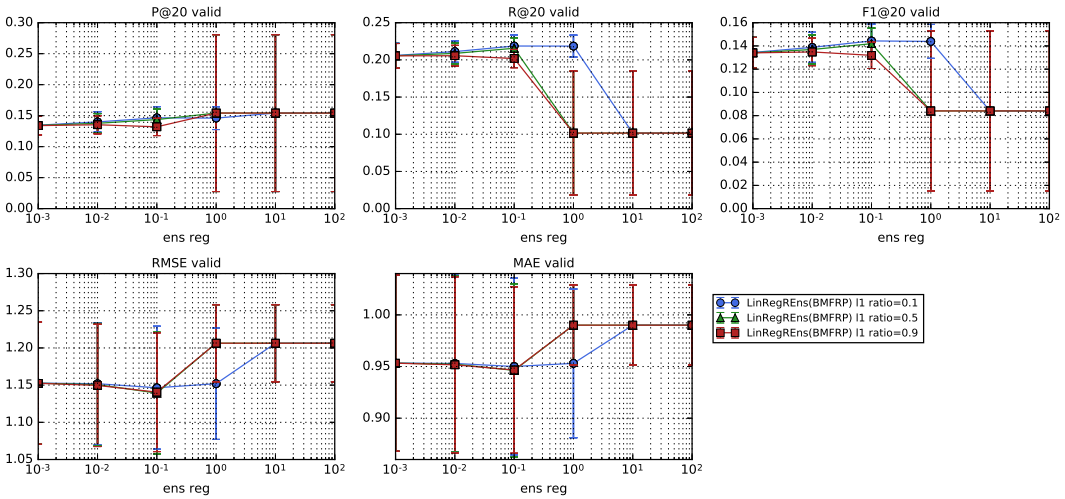


Figura 16 – Análise dos parâmetros de regularização para o *ensemble* (validação). No eixo x tem-se o coeficiente de regularização λ (ens reg) e no eixo y as métrcas para diferentes valores de α (l1 ratio).

Tabela 2 – Movielens – Tempos de treinamento (excluindo tempo para MF) em segundos (média e desvio padrão na validação), junto com as principais métricas de acurácia. O tempo do *ensemble* é reportado como $\Delta + T$, onde Δ é o tempo de treinamento dos modelos de base e T o tempo de treinamento do *ensemble*. O método BMF é o proposto na literatura, enquanto os demais são propostos neste presente trabalho.

Método	Tempo de treinamento (s)	P@20	RMSE
$l = \|U\|$			
BMF ($m = 100\%$)	**34,7 ± 0,1**	0,086 ± 0,011	**1,014 ± 0,004**
BMFRP ($m = 100\%, d = 50\%$)	39,7 ± 1,0	0,097 ± 0,012	1,015 ± 0,003
BMFRP ($m = 100\%, d = 40\%$)	37,0 ± 0,1	0,103 ± 0,014	1,017 ± 0,004
BMFRP ($m = 100\%, d = 20\%$)	35,7 ± 0,1	0,100 ± 0,013	1,021 ± 0,003
BMF+LSH ($m = 100\%$)	55,3 ± 0,7	0,086 ± 0,011	**1,014 ± 0,004**
BMFRP+LSH ($m = 100\%, d = 50\%$)	47,2 ± 0,4	0,100 ± 0,015	1,016 ± 0,004
Ensemble BMFRP ($m = 100\%$)	Δ+ 0,3 ± 0,3	**0,144 ± 0,017**	1,139 ± 0,082
$l = 0,5\|U\|$			
BMF ($m = 100\%$)	**17,6 ± 0,1**	0,063 ± 0,009	**1,019 ± 0,004**
BMFRP ($m = 100\%, d = 50\%$)	18,7 ± 0,1	0,061 ± 0,008	1,021 ± 0,003
BMFRP ($m = 100\%, d = 40\%$)	19,8 ± 0,1	0,061 ± 0,009	1,020 ± 0,003
BMFRP ($m = 100\%, d = 20\%$)	18,7 ± 0,1	0,058 ± 0,008	1,023 ± 0,004
BMF+LSH ($m = 100\%$)	32,1 ± 3,0	0,047 ± 0,008	1,036 ± 0,004
BMFRP+LSH ($m = 100\%, d = 50\%$)	26,9 ± 2,2	0,052 ± 0,009	1,036 ± 0,004
Ensemble BMFRP ($m = 100\%$)	Δ+ 0,3 ± 0,2	**0,122 ± 0,011**	1,139 ± 0,081
$l = 0,3\|U\|$			
BMF ($m = 100\%$)	**10,6 ± 0,0**	0,056 ± 0,011	1,026 ± 0,004
BMFRP ($m = 100\%, d = 50\%$)	11,8 ± 0,0	0,052 ± 0,012	1,028 ± 0,003
BMFRP ($m = 100\%, d = 40\%$)	12,9 ± 0,1	0,051 ± 0,007	1,030 ± 0,003
BMFRP ($m = 100\%, d = 20\%$)	11,7 ± 0,0	0,046 ± 0,007	1,031 ± 0,005
BMF+LSH ($m = 100\%$)	23,8 ± 1,2	0,047 ± 0,006	1,044 ± 0,006
BMFRP+LSH ($m = 100\%, d = 50\%$)	20,1 ± 1,1	0,042 ± 0,007	1,050 ± 0,005
Ensemble BMFRP ($m = 100\%$)	Δ+ 0,3 ± 0,2	**0,156 ± 0,009**	1,140 ± 0,081

Tabela 3 – Movielens – Comparação de métricas de acurácia para cada um dos métodos testados. O tempo do *ensemble* é reportado como $\Delta + T$, onde Δ é o tempo de treinamento dos modelos de base e T o tempo de treinamento do *ensemble*.

Method	Tempo de treinamento (s)	P@20	RMSE
BMF ($m = 100\%$, $l = \|U\|$)	1772,9 ± 350,1	0,086 ± 0,011	1,014 ± 0,004
BMFRP ($m = 100\%$, $d' = 50\%d$, $l = \|U\|$)	1777,9 ± 350,9	0,097 ± 0,012	1,015 ± 0,003
BMF+LSH ($m = 100\%$)	1793,5 ± 350,6	0,086 ± 0,011	1,014 ± 0,004
BMFRP+LSH ($m = 100\%, d = 50\%$)	1785,4 ± 349,8	0,100 ± 0,015	1,016 ± 0,004
Ensemble BMFRP ($m = 100\%$, $l = \|U\|$, $d' \in [0,1d; 0,3d]$)	Δ+0,3 ± 0,3	0,144 ± 0,017	1,139 ± 0,082
ItemBased	111,3 ± 0,4	0,078 ± 0,005	**0,932 ± 0,006**
UserBased	37,3 ± 1,4	0,045 ± 0,008	0,951 ± 0,007
SVD ($f = 363$)	**24,8 ± 0,3**	**0,336 ± 0,020**	0,952 ± 0,006

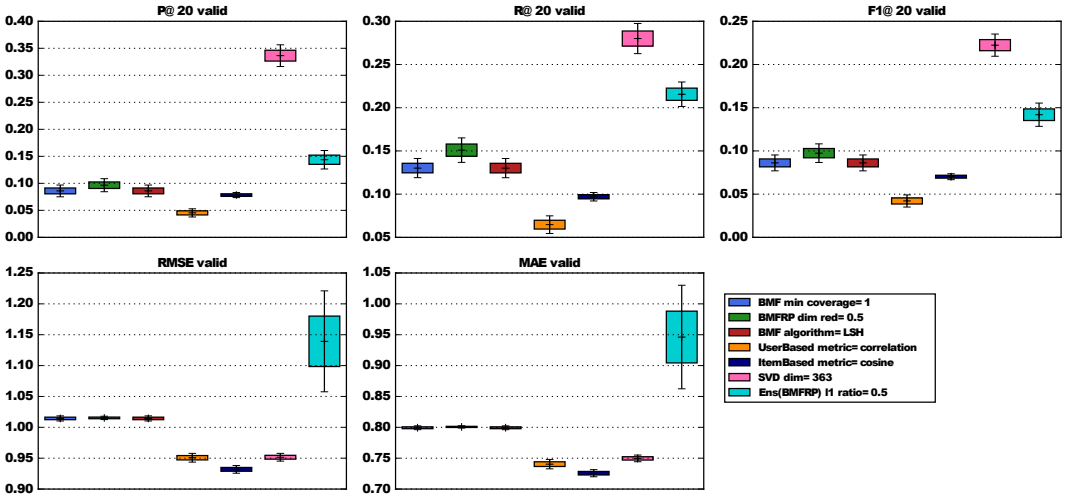


Figura 17 – Métricas de erro e lista para diferentes modelos propostos (validação). O título de cada gráfico indica a métrica considerada (eixo y) e cada bloco corresponde a um modelo (eixo x) indicado pela cor na legenda.

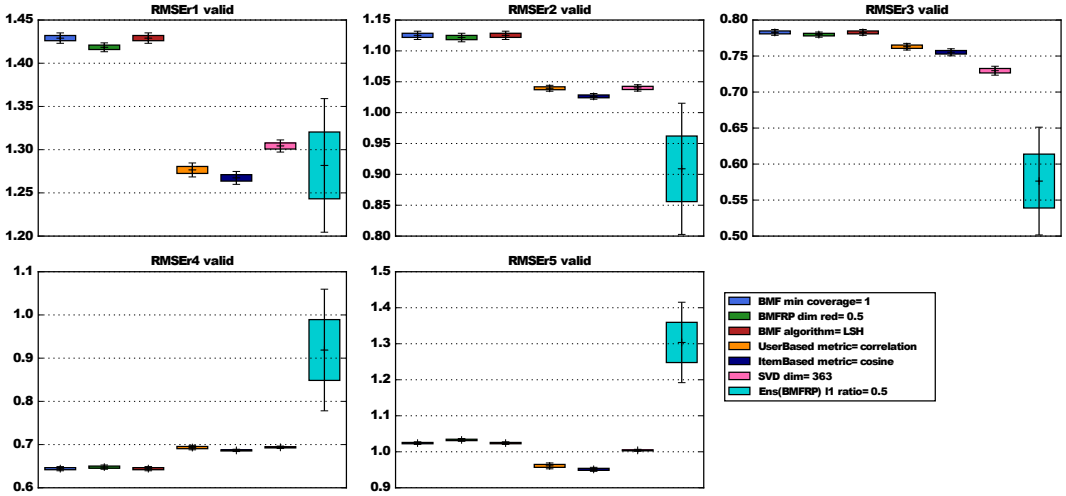


Figura 18 – Métricas de erro RMSE por nota para os diferentes modelos propostos (validação). O título de cada gráfico indica a nota de referência para o cálculo do RMSE (valores no eixo y) e cada bloco corresponde a um modelo (eixo x) indicado pela cor na legenda.

7.2.2 Experimentos na base Delicious2k

Nessa base, pode-se observar que o uso de coberturas menores não afeta o erro. Sua influência é percebida apenas numa variação na precisão (ver Figura 19). Sendo a base muito grande para experimentos com validação cruzada em 5 pastas, foi feita apenas uma validação *holdout*. Como não foi feita uma validação em várias pastas, obtendo-se uma estimação do desempenho médio nas várias métricas, é natural que tenha havido uma variação sem tendência definida, que pode ser devida a particularidades da partição de dados considerada.

Além disso, nesse caso pode-se notar que o uso da BMF já traz uma redução de dimensão bastante expressiva, mesmo utilizando a cobertura total. A matriz de dados original, com mais de 30 mil colunas, pode ser representada em uma decomposição com apenas 2236

fatores — uma redução de 95%. Ainda assim o uso de coberturas menores é relevante, devido ao alto custo computacional de buscar todos os coclusters correspondentes aos fatores da decomposição.

Quanto ao tamanho do grafo kNN, dessa vez houve um comportamento diferente, com o erro constante após $l = 0,3|U|$, e métricas de lista mais altas para valores menores de l (ver Figura 20). Dessa forma, uma boa escolha seria $l = 0,3|U|$.

O uso de LSH nessa base trouxe um ganho no tempo de treinamento, possivelmente porque a base tem um tamanho maior e o *overhead* da construção do índice do LSH forest foi compensado pelo ganho em rapidez na busca de vizinhos (ver Tabela 5). Também não há significativa variação de acurácia, sendo que aumentos maiores no erro ocorrem para $l < |U|$.

A variação do número de candidatos não influenciou na acurácia, considerando diferentes tamanhos de modelo kNN (ver Figura 21). Já a variação do número de estimadores tem influência para casos com tamanho de modelo kNN menor — $l = 0,3|U|$ e $l = 0,5|U|$ (ver Figura 22). Os valores da Tabela 5 se referem a $2l$ candidatos e 10 estimadores, que é suficiente para $l = |U|$, mas piora a acurácia para $l < |U|$.

O uso de projeções aleatórias com reduções mais expressivas (ex: 30%, Tabela 5) chegaram a compensar em termos de redução do tempo de treinamento. Ainda assim, usar uma cobertura mínima menor (ex: 60%) leva a uma dimensão similar e tempo de treinamento equivalente. Considerando que uma cobertura menor implica num tempo de fatoração de matriz mais curto, esse método é mais eficaz em reduzir o tempo geral de treinamento.

Observando as performances, as precisões são mais ou menos as mesmas, havendo um erro ligeiramente menor para o uso de BMF+RP — comparando $m = 80\%$ com redução $d = 50\%$ e $m = 80\%$ com redução $d = 30\%$, que levam a dimensões próximas. A diferença no erro se torna maior quando são usados tamanhos de modelo kNN menores ($l = 0,5|U|$ e $l = 0,3|U|$).

No *ensemble* proposto, os parâmetros da *elastic net* não trouxeram variação de performance. Foram mantidos os mesmos valores utilizados para a base Movielens. Nessa base, obteve-se uma redução do erro concomitante com um aumento da precisão (ver Tabela 5).

Comparativamente a outros métodos, observa-se um cenário parecido com a base Movielens (ver Tabela 6. O SVD continua com a melhor performance, mas o *ensemble* proposto nesse caso teve a mesma performance que o SVD.

Tabela 4 – Base Delicious2k – Número de fatores encontrados para diferentes coberturas mínimas (média dos valores obtidos na validação), relacionados com o número de usuários $|U| = 1867$ e de itens $|I| = 38581$, e com o tempo gasto para calcular as matrizes.

Cobertura mínima	Número de fatores f	$f/\|U\|$	$f/\|I\|$	Tempo (h)
100%	2236	1,198	0,058	11,37
80%	1087	0,582	0,028	7,20
60%	711	0,381	0,018	3,97

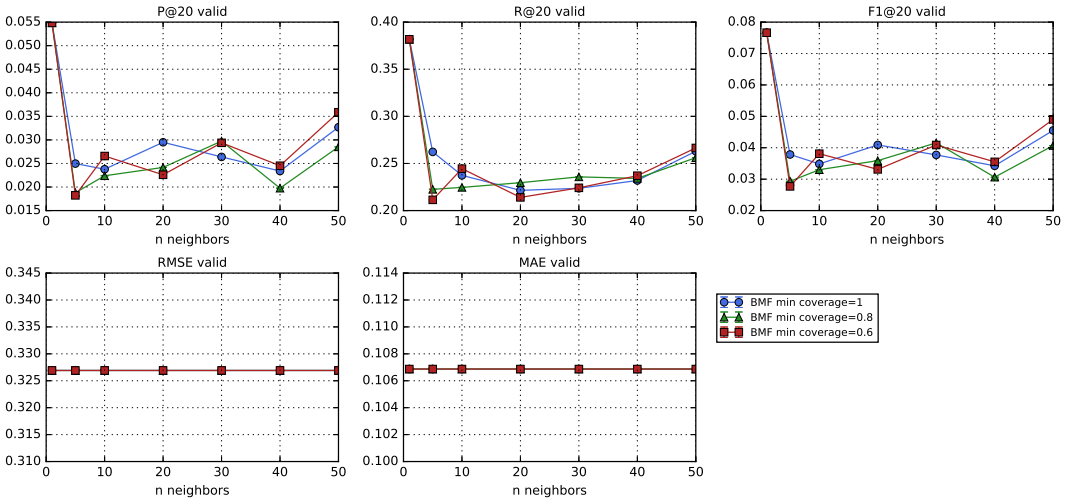


Figura 19 – Base Delicious 2k: Performance de diferentes coberturas mínimas, variando o número de vizinhos (validação).

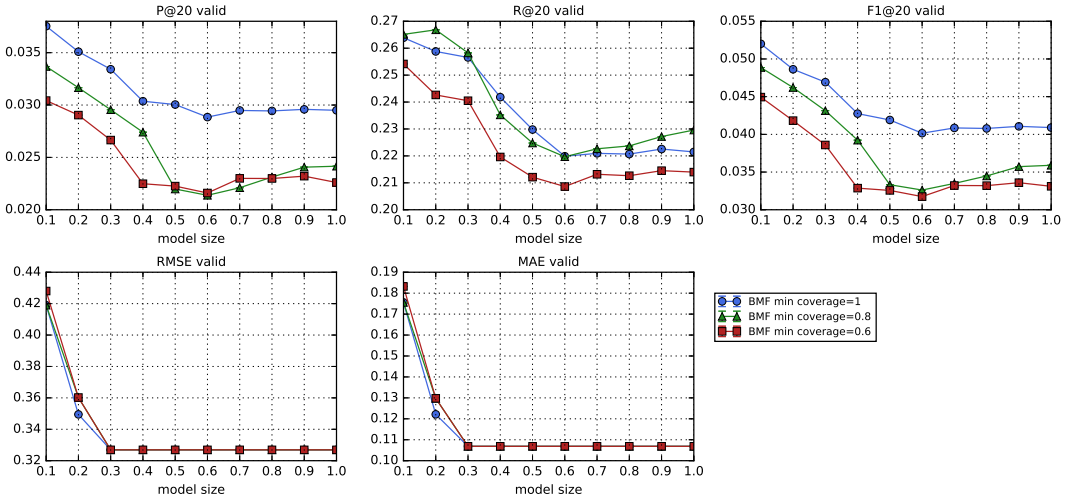


Figura 20 – Base Delicious 2k: Performance de diferentes coberturas mínimas, usando 20 vizinhos, variando o tamanho de modelo kNN (validação).

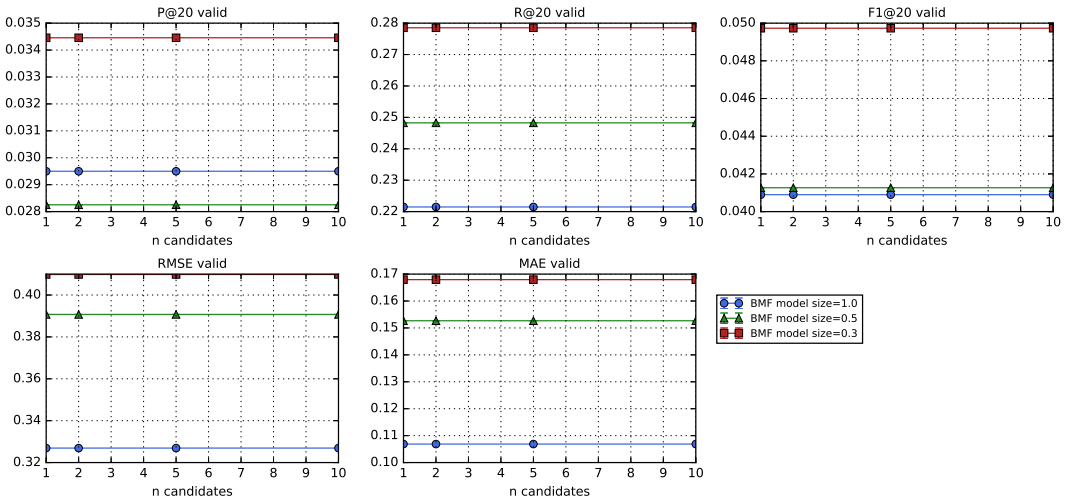


Figura 21 – Base Delicious2k: BMF com LSH, variando número de candidatos e tamanho de modelo kNN, usando 20 vizinhos. O numero de candidatos indicado corresponde apenas ao múltiplo n, de modo que o numero de candidatos de fato é nl.

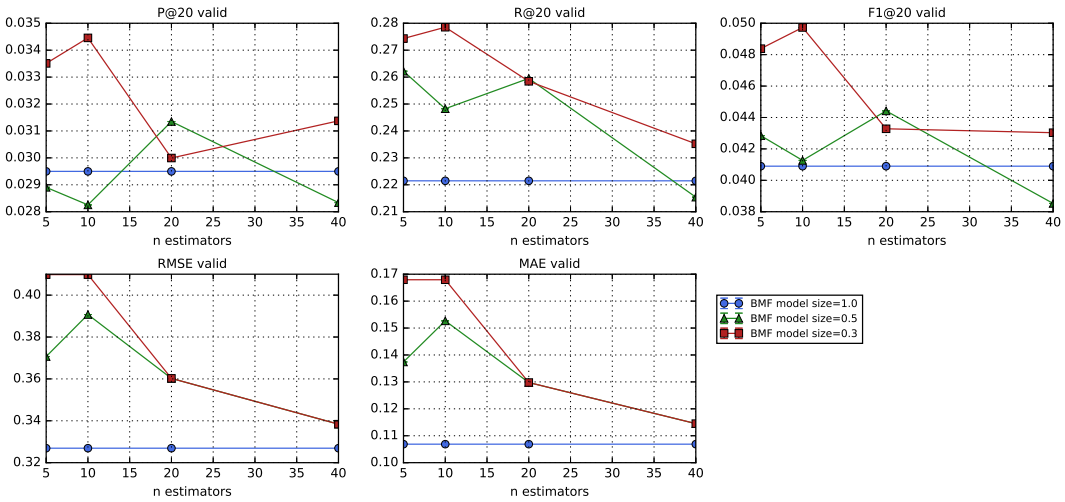


Figura 22 – Base Delicious2k: BMF com LSH, variando número de estimadores e tamanho de modelo kNN, usando 20 vizinhos.

Tabela 5 – Delicious 2k – Tempos de treinamento (excluindo tempo para MF) em segundos (média e desvio padrão na validação), junto com as principais métricas de acurácia. O tempo de treinamento do *ensemble* não considera o tempo de treinamento dos modelos de base. Foram usado 20 vizinhos. O método BMF é o proposto na literatura, enquanto os demais são propostos neste presente trabalho.

Método	Tempo treinamento (s)	P@20	RMSE
$l = \|U\|$			
BMF ($m = 100\%$)	148,0	0,02950	0,327
BMF ($m = 80\%$)	135,0	0,02416	0,327
BMFRP ($m = 100\%, d = 50\%$)	150,7	0,02484	0,311
BMF ($m = 60\%$)	134,8	0,02260	0,327
BMFRP ($m = 100\%, d = 30\%$)	136,2	0,02383	0,324
BMF+LSH ($m = 100\%$)	**132,9**	0,02950	0,327
Ensemble BMFRP ($m = 100\%$)	$\Delta+$ 0,6	**0,03883**	**0,000**
$l = 0,5\|U\|$			
BMF ($m = 100\%$)	82,8	0,03005	0,327
BMF ($m = 80\%$)	69,3	0,02198	0,327
BMFRP ($m = 100\%, d = 50\%$)	82,3	0,02309	0,265
BMF ($m = 60\%$)	75,8	0,02226	0,327
BMFRP ($m = 100\%, d = 30\%$)	**69,0**	0,02585	0,262
BMF+LSH ($m = 100\%$)	69,8	0,02825	0,391
Ensemble BMFRP ($m = 100\%$)	$\Delta+$ 0,7	**0,03883**	**0,000**
$l = 0,3\|U\|$			
BMF ($m = 100\%$)	45,3	0,03342	0,327
BMF ($m = 80\%$)	**42,7**	0,02955	0,327
BMFRP ($m = 100\%, d = 50\%$)	53,3	0,02616	0,290
BMF ($m = 60\%$)	46,6	0,02665	0,327
BMFRP ($m = 100\%, d = 30\%$)	48,4	0,02989	0,276
BMF+LSH ($m = 100\%$)	43,2	0,03446	0,410
Ensemble BMFRP ($m = 100\%$)	$\Delta+$ 0,6	**0,03883**	**0,000**

Tabela 6 – Delicious 2k – Comparação de métricas de acurácia para cada um dos métodos testados. O tempo do *ensemble* é reportado como $\Delta + T$, onde Δ é o tempo de treinamento dos modelos de base e T o tempo de treinamento do *ensemble*.

Method	Tempo de treinamento (s)	P@20	RMSE
BMF ($m = 100\%$, $l = \|U\|$)	138,6	0,03107	0,318
BMFRP ($m = 100\%$, $d' = 50\%d$, $l = \|U\|$)	140,5	0,02792	0,287
BMF+LSH ($m = 100\%$)	221,2	0,0305	0,318
BMFRP+LSH ($m = 100\%, d = 50\%$)	191,4	0,03051	0,266
Ensemble BMFRP ($m = 100\%$, $l = \|U\|$, $d' \in [0,1d; 0,3d]$)	Δ+0,7	**0,04984**	**0,000**
UserBased	193,8	0,00000	1,000
SVD ($f = 2236$)	**9,5**	**0,04984**	**0,000**

7.3 Considerações finais

Pode-se observar que, na base Movielens, o uso das técnicas baseadas em BMF apresenta resultados razoáveis em termos de RMSE, apesar da binarização da escala de notas. Há uma melhor precisão em relação aos métodos de vizinhança tradicionais IB e UB. Contudo, continua-se com uma performance inferior ao SVD, estado-da-arte. Na base Delicious observam-se resultados semelhantes. Contudo o *ensemble* proposto atinge uma performance máxima em RMSE, juntamente com o SVD.

O uso da vizinhança aproximada LSH não degrada a performance mas tampouco melhora o tempo de treinamento, para a base Movielens. Já para a base Delicious, ele chega a diminuir o tempo em alguns casos. Isso provavelmente se deve ao fato da base ser maior, e o *overhead* de construção dos índices LSH ser compensado pela aceleração do cálculo dos vizinhos. Algo similar ocorre com o uso de projeções aleatórias. Nesse caso, acelerações maiores podem ser obtidas com implementações mais eficientes que se utilizem da estrutura esparsa das matrizes de projeção.

O *ensemble* proposto traz melhoras de precisão significativas em relação aos modelos BMF de base, com pouco overhead de tempo de treinamento, se for considerado que os modelos de base possam ser treinados em paralelo. Ele atinge performances mais comparáveis às do SVD (mesma ordem de grandeza) e na base Delicious teve o mesmo desempenho.

8 Coclusterização contígua em dados de atividade cerebral

Como dito anteriormente, a coclusterização contígua é uma técnica útil para a descoberta de padrões coerentes co-ocorrentes em séries temporais simultâneas. Esse método foi aplicado com sucesso na literatura para dados de expressão gênica, e foi sugerido que sua aplicação para outras séries biológicas é promissora. Em parceria com colegas do projeto CEPID/BRAINN, o algoritmo CCC-biclustering foi aplicado a dados de séries temporais cerebrais de fMRI (ressonância magnética funcional, do inglês *funcional magnetic resonance imaging*). O método também foi aplicado junto a dados abertos de EEG (eletroencefalograma). Esse algoritmo busca por coclusters contíguos, sendo bastante adaptado para detectar co-atividade entre séries temporais. Essa coclusterização encontra as séries que apresentam um comportamento coerente e também em quais janelas de tempo isso ocorre. Cada série corresponde à atividade de uma região do cérebro (ou eletrodo, no caso do EEG). Com isso identificamos regiões co-funcionais, ou seja, que apresentam padrões de atividade coerentes.

Propusemos o uso da informação obtida na coclusterização para a obtenção de uma representação na forma de um mapa funcional. As regiões de monitoramento são representadas por nós de uma rede e as conexões entre os nós indicam atividade cofuncional. Com isso, pode-se obter uma visualização mais intuitiva dos padrões encontrados. Também foi feita uma análise de algumas características topológicas dessas redes, buscando observar sua capacidade de discriminar indivíduos do grupo de controle e pacientes.

Na seção 8.1, é apresentada a metodologia desta linha de trabalho, detalhando os conjuntos de dados e as análises realizadas. Em seguida, a seção 8.2 apresenta e avalia os resultados obtidos. Considerações finais referentes a essa linha de trabalho são apresentadas na seção 8.3.

8.1 Metodologia

8.1.1 Dados

O algoritmo de coclusterização contígua foi implementado e aplicado em dois conjuntos de dados:

fMRI Esse conjunto corresponde a séries temporais de 90 regiões do cérebro, obtidas a partir de imagens de fMRI coletadas e pré-processadas por parceiros do BRAINN[1]. Os indivíduos em análise (19 pacientes e 19 controles) realizavam uma atividade motora, com

[1] Dados coletados por Raphael F. Casseb, do Laboratório de Neurofísica, FCM, Unicamp (experimento aprovado

pleno sincronismo temporal, não havendo, portanto, defasagem na atividade cognitiva. No pré-processamento, as imagens foram segmentadas em 90 regiões segundo um atlas cerebral padrão, calculando-se a série média dos voxels de cada uma.

EEG Esse conjunto de dados está disponível no *UCI Machine Learning Repository*[2], e corresponde a medições de EEG extra-craniano usando 64 eletrodos, por um período 1s a uma taxa de 256 Hz. Foram utilizadas séries referentes a um paciente que realizava uma tarefa motora. Foi feita a média das séries sobre 10 medições.

8.1.2 Pré e pós-processamento

As séries foram normalizadas, diferenciadas e quantizadas em três símbolos, D (desce), U (sobe) e N (neutro), conforme a proposta de Madeira *et al.* (2010) (seção 2.4). Esse pré-processamento implica na escolha de dois parâmetros: o atraso de diferenciação e o limiar de quantização, cujos valores padrão são ambos iguais a 1. Para o conjunto fMRI, o atraso na diferenciação foi de 5 amostras, devido à taxa de amostragem das séries e à informação a priori de que esperávamos padrões da ordem de 0,1 Hz. Quanto ao limiar, para o conjunto fMRI foi necessário utilizar um limiar mais baixo, de 0,01, para que fossem encontrados coclusters significativos. Para o conjunto EEG, foram utilizados o atraso e o limiar padrões. Após a coclusterização, foram mantidos apenas coclusters com mais de 20 colunas e que passaram no teste de significância estatística proposto por Madeira *et al.* (2010), com $p \leq 0,05$.

8.1.3 Visualização e análise

Seguindo o paradigma de modelagem do cérebro como uma rede complexa, a partir da informação trazida pela coclusterização contígua foi gerada uma visualização na forma de um grafo dinâmico. Com base no conjunto de coclusters obtido, foi gerada uma rede dinâmica onde seus nós representam regiões do cérebro (ou eletrodos de EEG) e suas arestas indicam que as regiões estavam juntas em um cocluster — apresentavam comportamento coerente — durante o período de tempo em que a aresta existir (Figura 23).

Inicialmente foram utilizadas arestas binárias. Mais tarde foi adicionada uma ponderação referente à correlação entre as regiões conectadas no período definido pelo cocluster correspondente.

As redes podem ser visualizadas com o software Gephi (BASTIAN *et al.*, 2009), que permite selecionar a janela de tempo desejada ou visualizar a evolução da rede em forma de vídeo (Figura 24). Ele permite ainda que os nós sejam posicionados de acordo com a localização anatômica das regiões do cérebro, o que foi adotado para as figuras seguintes. Todas as redes

pela Comissão de Ética da Unicamp); pré-processados por ele e por Luis C. T. Herrera, sob orientação da Profa. Gabriela Castellano, do Instituto de Física Gleb Wataghin (Unicamp).

[2] Disponibilizado em <https://archive.ics.uci.edu/ml/datasets/EEG+Database>, por Henri Begleiter do Neurodynamics Laboratory, State University of New York Health Center, Brooklyn.

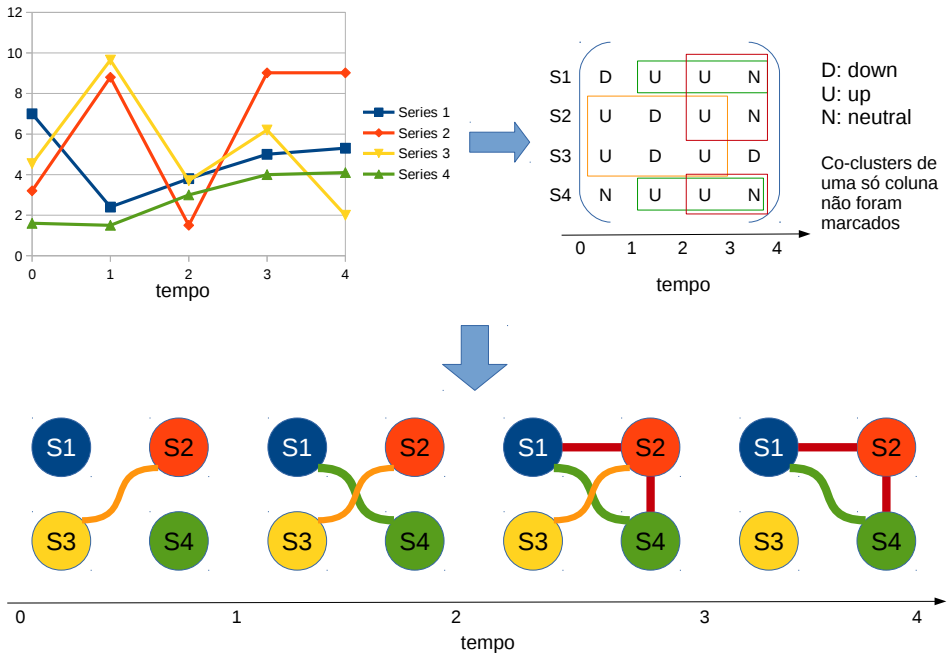


Figura 23 – Exemplo de geração de rede dinâmica a partir dos resultados da coclusterização contígua. Em cada instante de tempo tem-se uma rede com conexões diferentes, dependendo dos coclusters existentes. Primeiro apenas o cocluster conectando S2 e S3 existe. Em seguida, surge um cocluster conectando S1 e S4, depois o cocluster conectando S1, S2 e S4. Finalmente, cessa o cocluster ligando S2 e S3.

correspondem a uma vista superior do cérebro, com a parte anterior do cérebro mostrada em cima e a posterior em baixo.

Para o conjunto fMRI, onde vários indivíduos são considerados, duas abordagens foram tomadas:

- Foram geradas redes para cada um dos pacientes e controles. As redes de cada grupo foram combinadas, somando-se os pesos das arestas que se repetiram.
- Foi feita uma série média dos paciente e uma dos controles. A coclusterização contígua foi aplicada à série média, sendo obtida ao final uma rede para cada grupo.

8.1.4 Análise de métricas topológicas

A teoria de redes complexas conta com diversas métricas que quantificam características topológicas das redes, tanto em escala global como local. Algumas métricas, apresentadas no capítulo 6, foram selecionadas e computadas para as redes de cada indivíduo do conjunto fMRI:

- Densidade
- Eficiência global
- Clusterização média
- Centralidade de *betweeness*
- Centralidade de proximidade (*closeness*)

- Centralidade de grau

Foram utilizadas as redes de cada indivíduo, com as arestas ponderadas pela correlação entre as regiões conectadas no período definido pelo cocluster correspondente. Foram consideradas apenas correlações não-negativas, para simplificar a interpretação. Para verificar se as métricas distinguem controles e pacientes, um classificador foi treinado e avaliado em validação cruzada (10 pastas).

Diferentes métodos foram adotados na obtenção dos vetores de características utilizados como entrada do classificador, dependendo se são baseados em uma métrica local ou global. A saída do classificador é binária: indivíduo é controle ou paciente. A seguir é descrito como cada tipo de métrica é utilizada na geração de um vetor de características que representa cada indivíduo na classificação.

Para cada métrica global, foi calculado seu valor para cada instante, gerando um perfil temporal para cada indivíduo. Esse perfil temporal serve como o vetor de características representando cada indivíduo. Desse modo, a matriz de dados de entrada contém os 38 indivíduos nas linhas e os 110 instantes de tempo nas colunas. Em seguida, foi treinado um classificador kNN baseado nesses perfis, um para cada métrica. O classificador foi treinado com $k = 2$ vizinhos e utilizando a correlação de Pearson como métrica de similaridade.

Para as métricas locais, existe um perfil temporal por região do cérebro, para cada indivíduo. Foi calculada a média temporal para cada região, e cada indivíduo teve como vetor de características esse valor médio para cada região. Desse modo a matriz de dados de entrada contém os 38 indivíduos nas linhas e as 90 regiões do cérebro nas colunas. O uso de um classificador kNN não foi capaz de dar resultados razoáveis, por isso foi treinado um classificador SVM com kernel RBF e demais parâmetros no valor padrão (biblioteca Scikit-learn (PEDREGOSA *et al.*, 2011)), um para cada métrica.

Em ambos os cenários, foi calculada a taxa de falsos positivos e verdadeiros positivos em cada pasta de validação, sendo reportada a curva ROC (receiver operation characteristic) média de cada classificador.

8.2 Resultados e discussão

8.2.1 Experimentos iniciais

As primeiras análises foram feitas considerando-se um indivíduo de cada base. Na base fMRI, foram encontrados 109 biclusters, e na EEG, 120. As Figuras 25 e 27 mostram gráficos de alguns dos coclusters encontrados.

A quantidade de coclusters encontrados é relativamente alta para ser analisada diretamente através da visualização desses gráficos — especialmente se considerarmos que há

vários indivíduos a serem analisados. Essa é uma motivação a mais para a visualização em rede dinâmica. Ela permite ver quais regiões apresentaram comportamento coerente e em que momento. Com a biclusterização, é possível extrair esse comportamento localizado no tempo, não sendo necessário que as regiões estejam correlacionadas durante todo o período do experimento. As Figuras 26 e 28 mostram uma sequência temporal de imagens da rede dinâmica feita para um indivíduo de cada conjunto (EEG e fMRI, respectivamente), extraídas do software Gephi (Figura 24).

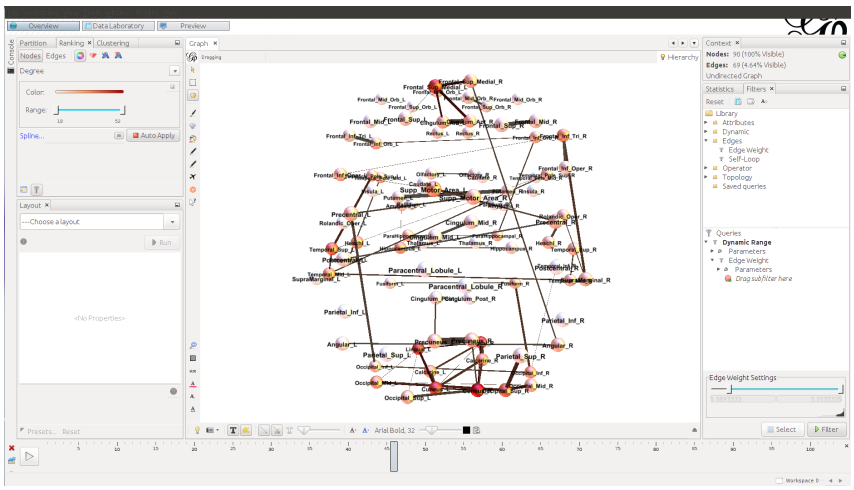


Figura 24 – Visualização da rede dinâmica no Gephi.

A rede do EEG é um pouco mais difícil de interpretar visualmente, uma vez que a atividade de uma certa região do cérebro se reflete em vários eletrodos, gerando um grande número de correlações e, consequentemente, arestas no grafo. Na rede do fMRI, a visualização é mais clara. As regiões realçadas em vermelho são as regiões com maior número de conexões.

8.2.2 Análise das séries médias — fMRI

Após essa análise preliminar, procedeu-se uma análise de grupo no conjunto fMRI. A Tabela 7 mostra a quantidade de coclusters encontrados nos grupos de controle e pacientes. Na Figura 29, temos a evolução temporal das redes determinadas pelos coclusters nas séries médias de cada grupo. As regiões maiores e mais vermelhas são nós com maior grau (mais densamente conectados). Não fica clara uma distinção entre os períodos de movimento e repouso. Entretanto, é clara a diferença entre os dois grupos ao longo do experimento, com regiões mais densamente conectadas que aparecem em apenas um dos grupos. Foi feita análise similar considerando a combinação das redes individuais em cada grupo, com resultados similares. A principal diferença é que o número de arestas é bastante superior.

Tabela 7 – Número de coclusters encontrados no conjunto fMRI - análise de grupo

Grupo	Num. médio coclusters	Num. coclusters série média
Controles	107 ± 28	100
Pacientes	115 ± 32	162

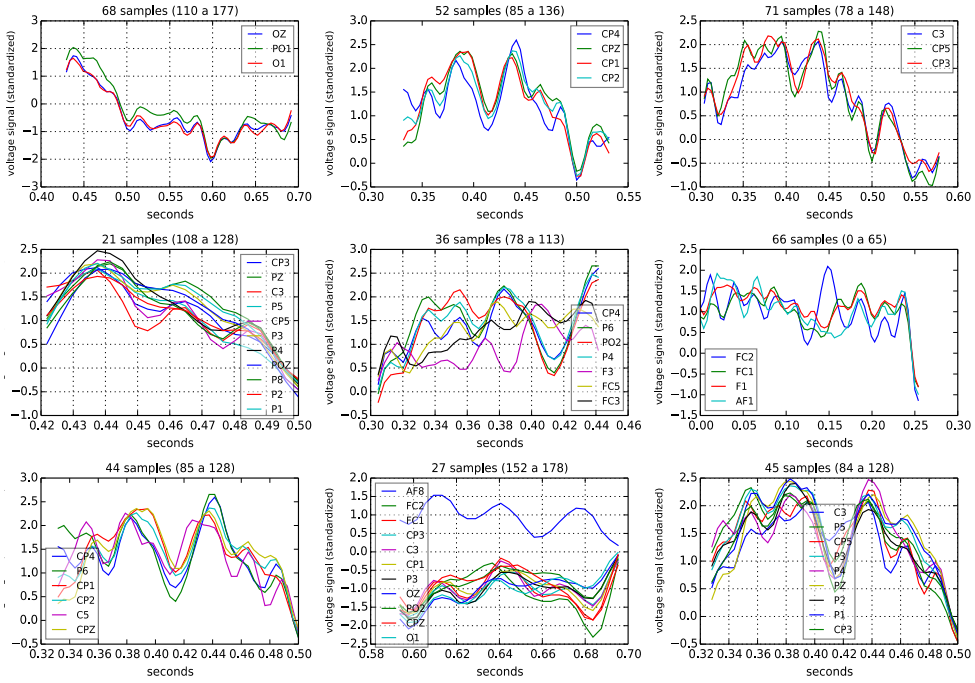


Figura 25 – Gráficos de alguns dos coclusters encontrados para um indivíduo do conjunto de dados EEG

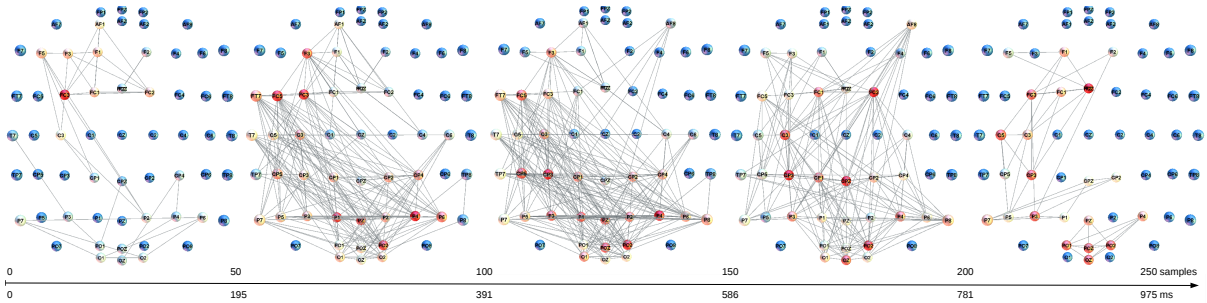


Figura 26 – Sequência temporal de imagens da rede dinâmica feita para um paciente do conjunto EEG.

8.2.3 Análise de métricas topológicas globais — fMRI

Nesse tipo de análise, é relevante observar as redes filtrando arestas com correlação muito baixa. Esse limiar t foi variado de 0 a 0,9 e os resultados podem ser vistos na Figura 30. Na média, pode-se perceber uma diferença entre controles e pacientes para $t \leq 0,8$. Nas três métricas, tem-se um comportamento aproximadamente constante para $t \leq 0,5$, que foi escolhido como limiar para as análises subsequentes.

Na Figura 31, pode-se ver o perfil temporal médio das métricas calculadas para cada grupo, com $t = 0,5$. Pode-se observar uma leve diferença entre controles e pacientes para as três métricas consideradas. A densidade e a clusterização média para o grupo de controle é menor que para o grupo de pacientes, enquanto a eficiência global é maior. Isso pode indicar que o cérebro dos pacientes recruta mais regiões e grupos de regiões durante a tarefa motora e ativa caminhos funcionais menos eficientes que os ativados por indivíduos sadios.

Esses resultados motivam o teste dessas métricas como fator discriminante entre

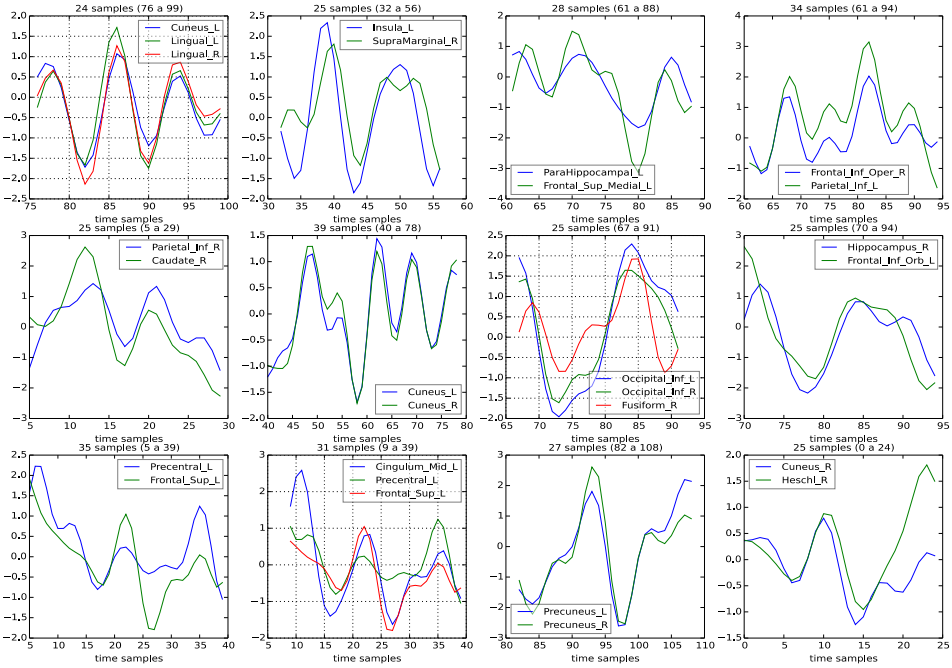


Figura 27 – Gráficos de alguns dos coclusters encontrados para um paciente do conjunto fMRI

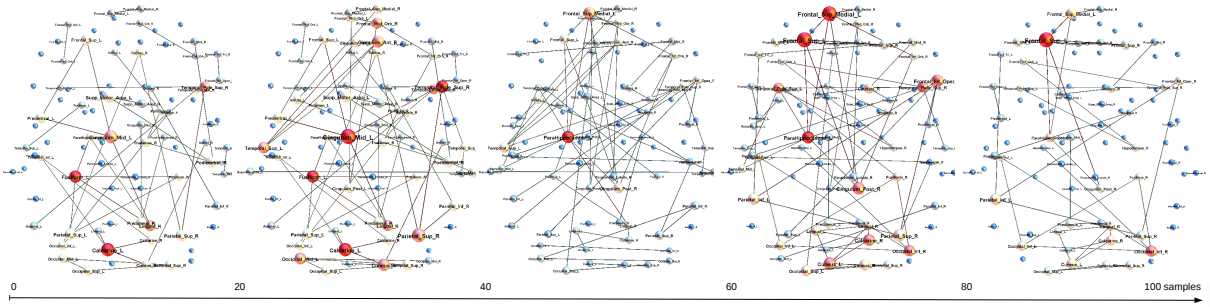


Figura 28 – Sequência temporal de imagens da rede dinâmica feita para um paciente do conjunto fMRI.

controles e pacientes. Para isso, foi treinado um classificador kNN para cada uma das métricas. Na Figura 32, tem-se o desempenho dos três classificadores. Pode-se observar que o uso da eficiência global como atributo levou a um desempenho razoável e bastante superior ao das outras métricas, indicando que ele foi o mais capaz de discriminar os grupos. Já a densidade e a clusterização média têm desempenhos fracos, próximos de um classificador aleatório. Como comentado na seção 6.1, isso reflete a capacidade do cérebro de combinar rapidamente informações de regiões distribuídas. Pode-se levantar a hipótese que o dano cerebral dos pacientes faz com que caminhos mais longos sejam utilizados no processamento da tarefa motora, reduzindo sua eficiência global significativamente em relação a indivíduos sadios.

8.2.4 Análise de métricas topológicas locais — fMRI

Na análise de métricas locais — centralidade de grau, de proximidade e *betweeness* — havia um perfil temporal para cada região. Foi tomada a média temporal para cada uma e,

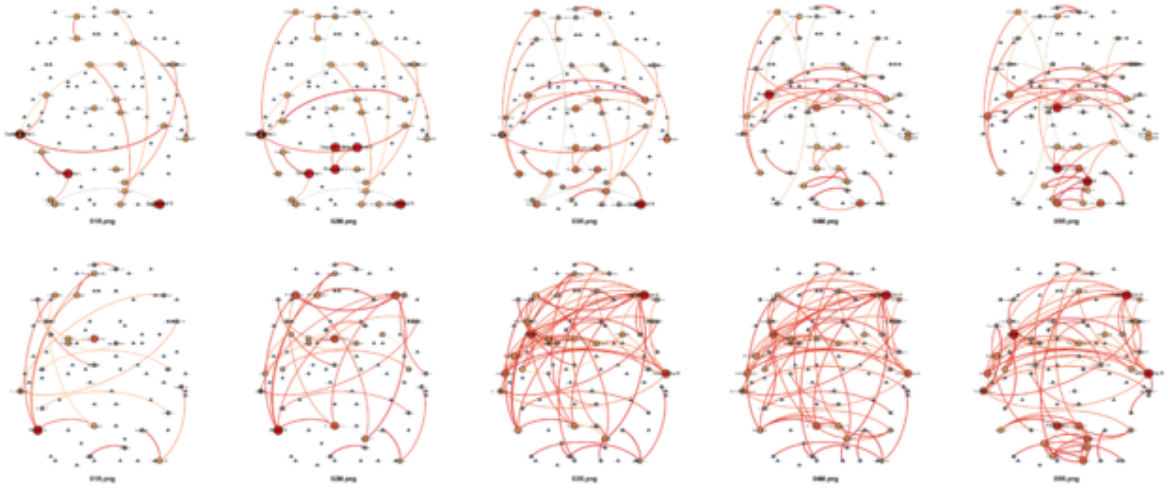


Figura 29 – Visualização das redes segundo intervalos de repouso e atividade seguidos no protocolo de coleta das séries, para controles (acima) e pacientes (abaixo). São mostrados apenas os primeiros 5 intervalos (na ordem repouso-movimento). Redes baseadas nas séries médias de cada grupo, com arestas ponderadas pela correlação entre a atividade das duas regiões no período de tempo indicado pelo cocluster.

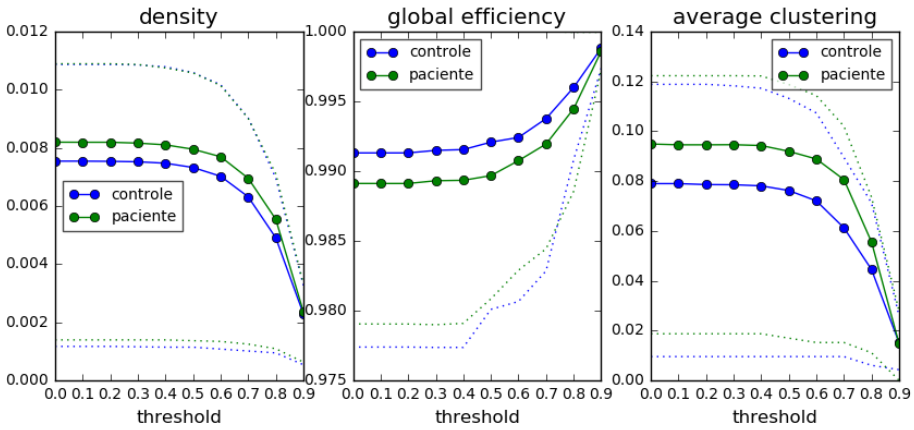


Figura 30 – Valor médio das métricas topológicas para controles e pacientes, com a rede filtrada segundo diferentes limiares de correlação. O perfil temporal de cada indivíduo foi agregado com a média. A linha cheia indica o valor médio sobre os indivíduos de cada grupo e as linhas pontilhadas indicam o máximo e o mínimo, a fim de se ter uma noção da amplitude de variação de cada uma das métricas.

em seguida, feita uma seleção de variáveis baseada no teste ANOVA, para escolher as regiões cujas métricas melhor discriminassem as classes. Variáveis com $p < 0,05$ foram selecionadas. Em seguida, foi treinado um classificador SVM, um para cada métrica testada. A curva ROC para os 3 classificadores pode ser vista na Figura 33.

As diferentes medidas testadas tiveram um poder de discriminação baixo, sendo a melhor performance obtida com a centralidade de *betweeness*. Isso indica que não há grande diferença entre os grupos nas características de centralidade utilizadas na classificação. O melhor desempenho da centralidade de *betweeness* pode indicar que a diferença mais relevante é quanto aos nós *hub*, pelos quais passam os caminhos mais curtos.

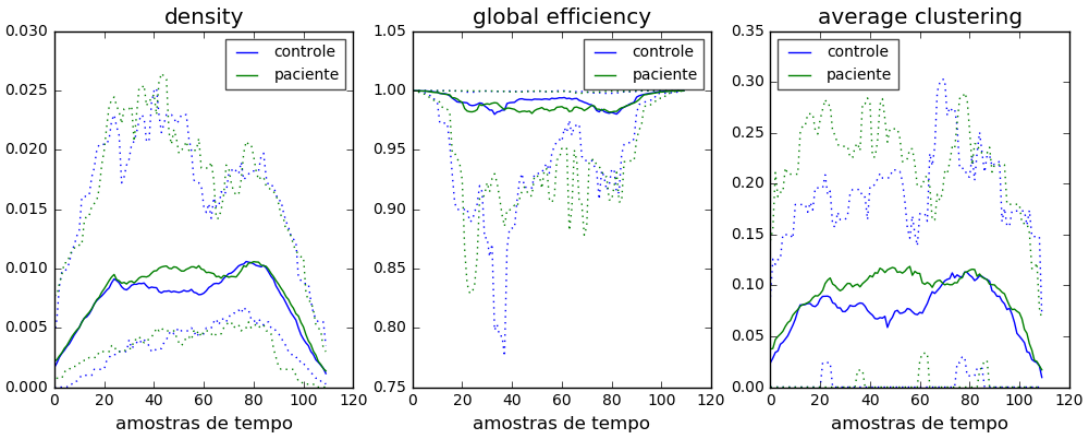


Figura 31 – Perfil temporal de métricas topológicas calculado para controles e pacientes. A linha cheia indica a média dos perfis de cada grupo, e as linhas pontilhadas indicam o máximo e o mínimo, a fim de se ter uma noção da amplitude de variação de cada uma das métricas.

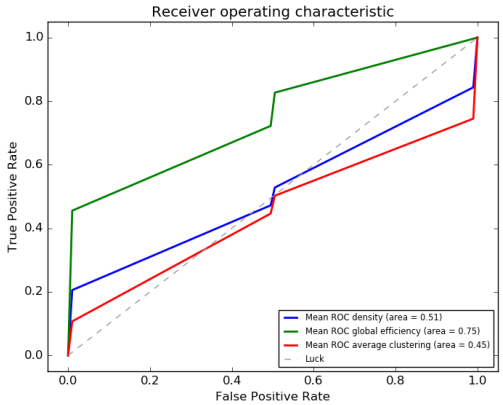


Figura 32 – Desempenho dos classificadores obtidos usando cada uma das métricas globais como atributo. Quanto maior a área sob a curva, melhor o desempenho.

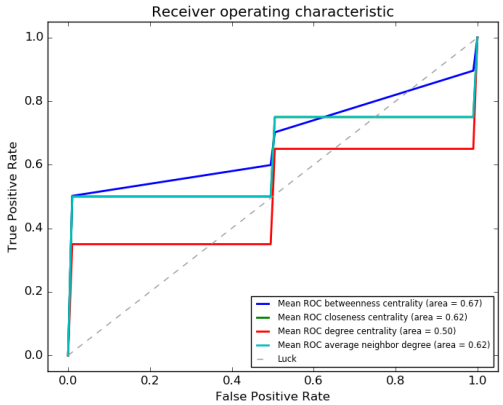


Figura 33 – Desempenho dos classificadores obtidos usando cada uma das métricas locais como atributo. Quanto maior a área sob a curva, melhor o desempenho.

8.3 Considerações finais

Nessa linha de trabalho foi explorada a coclusterização contígua como técnica de reconhecimento de padrões em dados de atividade cerebral. Os coclusters encontrados indicam séries temporais com atividade coerente durante uma janela de tempo contígua. Os coclusters

são maximais, de modo que a maior janela de tempo contígua é retornada, e todas as séries que apresentam comportamento coerente naquela janela fazem parte do cocluster. Essa característica é interressante para a segunda parte da proposta, que é a utilização dos coclusters na geração de uma representação em rede dinâmica do cérebro. Pode-se observar uma diferença entre as redes médias do grupo de controle e do grupo de pacientes. Foram realizados experimentos de classificação baseados em diversas métricas de rede, buscando validar a capacidade das redes de diferenciar os dois grupos. A métrica de eficiência global foi a que apresentou melhor capacidade de discriminação.

9 Conclusão

Neste trabalho, foram estudadas e aplicadas duas técnicas oriundas da coclusterização: a fatoração booleana de matrizes e a coclusterização contígua. Dois domínios de aplicação distintos foram considerados: a filtragem colaborativa para sistemas de recomendação e a análise de dados de atividade cerebral.

A fatoração booleana encontra uma decomposição de uma matriz binária em um produto booleano de matrizes, através da busca de coclusters binários de valores constantes. Essa técnica foi utilizada na síntese de filtros colaborativos por Ignatov *et al.* (2014), trabalho que foi estendido aqui através da proposta de um *ensemble* de projeções aleatórias.

Já a coclusterização contígua encontra coclusters de colunas contíguas cujas linhas apresentem padrões de evolução coerentes. O método proposto por Madeira *et al.* (2010) se baseia na construção de uma árvore de sufixos, e encontra todos os coclusters maximais em tempo polinomial. Neste trabalho, essa técnica foi aplicada no contexto de análise de dados de atividade cerebral.

A seguir, são apresentadas as conclusões e perspectivas futuras para cada uma das abordagens propostas neste trabalho.

9.1 Comitês de filtros colaborativos baseados em BMF e RP

Nesta frente de trabalho, foi proposta uma extensão de um *framework* de recomendação que usa fatoração booleana de matrizes, através da agregação de múltiplas projeções aleatórias das matrizes de fatores em um ensemble. Também foi estudado o uso de uma técnica de vizinhança aproximada baseada em LSH.

O uso da fatoração booleana em filtragem colaborativa apresenta resultados razoáveis em termos de RMSE, mesmo com a perda de informação acarretada pela binarização da escala de notas. O método apresenta melhor precisão que métodos tradicionais como IB e UB, apesar de ainda ser inferior ao SVD.

Como o cálculo da fatoração booleana envolve a busca de coclusters, trata-se de um problema NP-completo e de custo computacional elevado. Contudo, não é necessário buscar todos os coclusters para se atingir uma aproximação razoável. Nos experimentos realizados, uma busca de conceitos de modo a cobrir 80% da matriz original é significativamente mais rápido, e traz pouca degradação no erro e praticamente nenhuma perda de precisão. Além disso, o uso de uma cobertura parcial traz uma redução de dimensão que pode acelerar as etapas subsequentes, como a busca de vizinhos para a recomendação.

O uso do método de vizinhança aproximada praticamente não degrada os resultados, e aumentou um pouco o tempo de treinamento. Possivelmente, o ganho de velocidade no cálculo dos vizinhos foi menor que o custo da construção do índice LSH, para o tamanho da base de dados considerada. Em bases de dados maiores, esse *overhead* pode ser menos significativo, e o uso de LSH para vizinhança pode compensar e trazer ganhos efetivos.

O uso de projeções aleatórias foi capaz de reduzir a dimensão com pouco acréscimo no tempo de treinamento, basicamente mantendo a performance. Implementações mais eficientes, tirando proveito da estrutura esparsa das matrizes de projeção e de dados, podem ajudar a reduzir esse *overhead*. Para conjuntos de dados maiores, é possível que esse *overhead* seja menos significativo, e seja observado de fato um ganho no tempo de treinamento.

Os experimentos mostram que há uma grande variabilidade nos resultados obtidos após projeção aleatória (grande desvio padrão nas métricas calculadas), sugerindo que múltiplas projeções de fato geram diversidade para serem combinadas em um comitê. Foi testada a agregação de 11 projeções entre 10% e 30% da dimensão. Também poderia ser estudada a influência do uso de uma maior quantidade de projeções, em outras faixas de dimensão. O *ensemble* proposto apresentou melhor precisão que os métodos utilizando apenas BMF, BMF+RP ou BMF+LSH, e também melhor que os sistemas IB e UB.

O comitê proposto apresentou robustez ao uso de tamanhos de modelo kNN menores, mantendo boa precisão, o que permite acelerar o treinamento dos modelos de base sem perda de performance. É importante ressaltar que, devido ao baixo custo da realização de projeções aleatórias, a geração do *ensemble* é pouco custosa, desde que o cálculo dos grafos kNN subsequentes possam acontecer em paralelo.

9.2 Coclusterização contígua em dados de atividade cerebral

Essa linha de trabalho consistiu em um estudo exploratório propondo o uso da coclusterização contígua como técnica de reconhecimento de padrões aplicada a dados de atividade cerebral. Propôs-se também o uso da informação gerada pela coclusterização na geração de uma rede de conectividade funcional dinâmica. Nesse contexto, a coclusterização é um método que ao mesmo tempo em que busca correlações entre regiões, também determina a janela onde a correlação existe. Dessa forma, pode ser vista como uma alternativa a métodos de janelamento tradicionais.

Para validar a relevância das redes obtidas com a coclusterização, foram realizados alguns experimentos de classificação de controles e pacientes, utilizando o perfil temporal de diversas métricas topológicas como atributo. Para algumas métricas, obteve-se um desempenho razoável, indicando que as redes são discriminantes entre os dois grupos. Para uma validação completa do método, seria relevante comparar as redes obtidas com redes geradas aleatoriamente, a fim de ver se as diferenças observadas não são fruto de mero acaso. Também seria

relevante aplicar a metodologia a outros conjuntos de dados, com pacientes sob outras condições, realizando outros tipos de tarefa ou em repouso. Num ponto de vista biológico, seria interessante verificar se as regiões mais participativas da rede são de fato regiões que normalmente participam da tarefa considerada, seja ela visual, motora, ou ambas.

Na análise de EEG é necessário maior pre-processamento das séries antes da aplicação da coclusterização. Pode-se filtrar diferentes bandas buscando-se analisar separadamente a atividade em cada uma. Pode ser interessante aplicar algum método não-supervisionado de separação de fontes (ex: ICA) a fim de identificar as fontes dos sinais medidos pelos sensores e, assim, determinar coclusters e redes mais significativos.

Tanto para os dados de EEG quanto para os de fMRI, pode ser interessante a seleção de componentes — eletrodos ou regiões — a serem considerados na análise. A seleção também pode ser realizada após a aplicação de ICA. Pode-se buscar componentes relevantes de maneira supervisionada, ou simplesmente escolher aquelas que biologicamente façam sentido para a análise desejada e segundo a tarefa realizada (motora, visual, entre outras).

Um outro caminho de análise poderia passar pela agregação dos coclusters encontrados a fim de identificar estados que sejam recorrentes, seja em um mesmo indivíduo ou em vários indivíduos com uma mesma condição de saúde. A identificação de estados também poderia ocorrer após a geração da rede, agregando as múltiplas camadas do grafo dinâmico.

Referências

ACHLIOPTAS, D. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, Elsevier BV, v. 66, n. 4, p. 671–687, jun 2003. ISSN 00220000. Citado na página 46.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, v. 17, n. 6, p. 734–749, 2005. Citado na página 33.

ALQADAH, F.; REDDY, C. K.; HU, J.; ALQADAH, H. F. Biclustering neighborhood-based collaborative filtering method for top-n recommender systems. *Knowledge and Information Systems*, Springer, v. 44, n. 2, p. 475–491, aug 2014. ISSN 02193116. Citado na página 35.

ANDONI, A.; INDYK, P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, ACM, New York, NY, USA, v. 51, n. 1, p. 117, jan 2008. ISSN 00010782. Citado na página 41.

AVOGADRI, R.; VALENTINI, G. Fuzzy ensemble clustering based on random projections for DNA microarray data analysis. *Artificial Intelligence in Medicine*, v. 45, n. 2-3, p. 173–183, feb 2009. ISSN 09333657. Citado na página 47.

BASSETT, D. S.; WYMBS, N. F.; PORTER, M. A.; MUCHA, P. J.; CARLSON, J. M.; GRAFTON, S. T. Dynamic reconfiguration of human brain networks during learning. In: *Proceedings of the National Academy of Sciences of the United States of America*. [S.l.]: National Academy of Sciences, 2011. v. 108, n. 18, p. 7641–7646. ISSN 0027-8424. Citado na página 52.

BASTIAN, M.; HEYMANN, S.; JACOMY, M. Gephi: An open source software for exploring and manipulating networks. In: *Third International AAAI Conference on Weblogs and Social Media*. [S.l.: s.n.], 2009. p. 361–362. ISBN 978-1-57735-421-5. ISSN 14753898. Citado na página 72.

BAWA, M.; CONDIE, T.; GANESAN, P. LSH forest. In: *Proceedings of the 14th international conference on World Wide Web - WWW '05*. New York, New York, USA: ACM Press, 2005. (WWW '05), p. 651. ISBN 1595930469. Citado 4 vezes nas páginas 18, 41, 55 e 60.

BELOHLAVEK, R.; VYCHODIL, V. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *Journal of Computer and System Sciences*, Elsevier Inc., v. 76, n. 1, p. 3–20, feb 2010. ISSN 00220000. Citado 2 vezes nas páginas 25 e 54.

BERTONI, A.; VALENTINI, G. Ensembles based on random projections to improve the accuracy of clustering algorithms. In: . [S.l.]: Springer Berlin Heidelberg, 2006. p. 31–37. ISBN 978-3-540-33183-4, 978-3-540-33184-1. Citado na página 47.

BINGHAM, E.; MANNILA, H. Random projection in dimensionality reduction. In: ACM. *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, New York, USA: ACM Press, 2001. p. 245–250. ISBN 158113391X. Citado 2 vezes nas páginas 45 e 46.

BOUCHER-RYAN, P. du; BRIDGE, D. Collaborative recommending using formal concept analysis. *Knowledge-Based Systems*, v. 19, n. 5, p. 309–315, sep 2006. ISSN 09507051. Citado na página 35.

BULLMORE, E.; SPORNS, O. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience*, Nature Publishing Group, v. 10, n. 3, p. 186–198, mar 2009. ISSN 1471-003X. Citado na página 48.

BUSYGIN, S.; BOYKO, N.; PARDALOS, P. M.; BEWERNITZ, M.; GHACIBEH, G.; SEREF, O.; KUNDAKCIOGLU, O. E.; PARDALOS, P. Biclustering EEG data from epileptic patients treated with vagus nerve stimulation. In: *AIP Conference Proceedings*. Gainesville, FL: AIP, 2007. v. 953, n. 1, p. 220–231. ISSN 0094243X. Citado na página 30.

CALHOUN, V. D.; ADALI, T. Time-varying brain connectivity in fMRI data: Whole-brain data-driven approaches for capturing and characterizing dynamic states. *IEEE Signal Processing Magazine*, IEEE, v. 33, n. 3, p. 52–66, may 2016. ISSN 1053-5888. Citado 2 vezes nas páginas 51 e 52.

CALHOUN, V. D.; MILLER, R.; PEARLSON, G.; ADALI, T. The chronnectome: Time-varying connectivity networks as the next frontier in fMRI data discovery. *Neuron*, v. 84, n. 2, p. 262–274, 2014. ISSN 10974199. Citado 2 vezes nas páginas 51 e 52.

CANDILLIER, L.; JACK, K.; FESSANT, F.; MEYER, F. State-of-the-art recommender systems. In: *Collaborative and Social ...*. [S.l.: s.n.], 2009. p. 1–22. Citado 3 vezes nas páginas 33, 35 e 36.

CHEN, C.; LI, D.; ZHAO, Y.; LV, Q.; SHANG, L. WEMAREC : Accurate and scalable recommendation through weighted and ensemble matrix approximation categories and subject descriptors. In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, New York, USA: ACM Press, 2015. p. 303–312. ISBN 9781450336215. Citado na página 38.

CHEN, J.; FANG, H.-r.; SAAD, Y. Fast approximate knn graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research*, JMLR, v. 10, n. 2009, p. 1989–2012, 2009. ISSN 1532-4435. Citado na página 43.

CHENG, Y.; CHURCH, G. M. Biclustering of expression data. In: *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*. [S.l.]: AAAI Press, 2000. p. 93–103. ISBN 1-57735-115-0. Citado na página 21.

CHIANG, S.; CASSESE, A.; GUINDANI, M.; VANNUCCI, M.; YEH, H. J.; HANEEF, Z.; STERN, J. M. Time-dependence of graph theory metrics in functional connectivity analysis. *NeuroImage*, v. 125, p. 601–615, 2016. ISSN 10959572. Citado na página 52.

COELHO, G. P. *Geração, Seleção e Combinação de Componentes para Ensembles de Redes Neurais Aplicadas a Problemas de Classificação*. Tese (Master) — Universidade Estadual de Campinas, 2006. Citado na página 37.

DASGUPTA, S.; GUPTA, A. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Struct. Alg.*, Wiley-Blackwell, v. 22, n. 1, p. 60–65, nov 2002. ISSN 1098-2418. Citado na página 46.

DECOSTE, D. Collaborative prediction using ensembles of maximum margin matrix factorizations. In: *Proceedings of the 23rd international conference on Machine learning*. [S.l.: s.n.], 2006. p. 249–256. Citado na página 38.

DONG, W.; MOSES, C.; LI, K. Efficient k-nearest neighbor graph construction for generic similarity measures. In: ACM. *Proceedings of the 20th international conference on World wide web*. New York, New York, USA: ACM Press, 2011. p. 577–586. ISBN 9781450306324. Citado na página 43.

DRUMOND, T. F.; VON ZUBEN, F. J. Ensemble for collaborative filtering based on random projections and Boolean matrix factorization. In: *Modelling, Simulation and Identification / 841: Intelligent Systems and Control*. Calgary,AB,Canada: ACTAPRESS, 2016. p. 281–288. ISBN 978-0-88986-983-7. Citado na página 54.

FALLANI, F. D. V.; ASTOLFI, L.; CINCOTTI, F.; MATTIA, D.; MARCIANI, M. G.; TOCCI, A.; SALINARI, S.; WITTE, H.; HESSE, W.; GAO, S.; COLOSIMO, A.; BABILONI, F. Cortical network dynamics during foot movements. *Neuroinformatics*, Humana Press Inc, Rome, v. 6, n. 1, p. 23–34, mar 2008. ISSN 1539-2791. Citado na página 52.

FALLANI, F. D. V.; RICHIARDI, J.; CHAVEZ, M.; ACHARD, S. Graph analysis of functional brain networks: practical issues in translational neuroscience. *Philosophical Transactions of the Royal Society B: Biological Sciences*, The Royal Society, v. 369, n. 1653, p. 1–12, sep 2014. ISSN 0962-8436. Citado 4 vezes nas páginas , 48, 49 e 52.

FERN, X. Z.; BRODLEY, C. E. Random projection for high dimensional data clustering:a cluster ensemble approach. In: FAWCETT, T.; MISHRA, N. (Ed.). *Proceedings of The Twentieth International Conference on Machine Learning*. Washington, DC: AAAI Press, 2003. p. 186–193. Citado na página 47.

FERN, X. Z.; BRODLEY, C. E.; OTHERS. *Cluster ensembles for high dimensional clustering: an empirical study*. [S.l.], 2006. Citado 3 vezes nas páginas 18, 47 e 54.

FREUND, Y.; IYER, R.; SCHAPIRE, R. E.; SINGER, Y. An efficient boosting algorithm for combining preferences. *The Journal of machine learning research*, JMLR. org, v. 4, p. 933–969, 2003. Citado na página 38.

FREUND, Y.; SCHAPIRE, R. A desicion-theoretic generalization of on-line learning and an application to boosting. *Computational learning theory*, v. 55, p. 119–139, 1995. ISSN 00220000. Citado na página 38.

GANTER, B.; STUMME, G.; WILLE, R. *Formal Concept Analysis, Foundations and Applications*. [S.l.: s.n.], 2005. v. 3626. 349 p. ISSN 15562646. ISBN 3540278915. Citado na página 23.

GIONIS, A.; INDYK, P.; MOTWANI, R.; OTHERS. Similarity search in high dimensions via hashing. In: *Proceedings of the 25th International Conference on Very Large Data Bases*. [S.l.: s.n.], 1999. v. 99, p. 518–529. Citado na página 41.

HECHT-NIELSEN, R. Context vectors: General purpose approximate meaning representations self-organized from raw data. *Computational intelligence: Imitating life*, IEEE press, p. 43–56, 1994. Citado na página 45.

HUTCHISON, R. M.; WOMELSDORF, T.; ALLEN, E. A.; BANDETTINI, P. A.; CALHOUN, V. D.; CORBETTA, M.; Della Penna, S.; DUYN, J. H.; GLOVER, G. H.; GONZALEZ-CASTILLO, J.; HANDWERKER, D. A.; KEILHOLZ, S.; KIVINIEMI, V.; LEOPOLD, D. A.; PASQUALE, F. de; SPORNS, O.; WALTER, M.; CHANG, C. Dynamic functional connectivity: Promise, issues, and interpretations. *NeuroImage*, v. 80, n. 3, p. 360–378, oct 2013. ISSN 10538119. Citado 2 vezes nas páginas 51 e 52.

HYVÄRINEN, A.; OJA, E. Independent component analysis: Algorithms and applications. *Neural Networks*, v. 13, n. 4-5, p. 411–430, 2000. ISSN 08936080. Citado na página 52.

IGNATOV, D. I.; NENOVA, E.; KONSTANTINOVA, N.; KONSTANTINOV, A. V. Boolean matrix factorisation for collaborative filtering: An FCA-based approach. In: AGRE, G.; HITZLER, P.; KRISNADHI, A. A.; KUZNETSOV, S. O. (Ed.). *Artificial Intelligence: Methodology, Systems, and Applications: 16th International Conference, AIMSA 2014, Varna, Bulgaria, September 11-13, 2014. Proceedings*. Cham: Springer International Publishing, 2014. p. 47–58. ISBN 978-3-319-10554-3. Citado 7 vezes nas páginas 18, 36, 37, 54, 55, 58 e 81.

INDYK, P.; MOTWANI, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In: *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*. New York, NY, USA: ACM, 1998. (STOC '98), p. 604–613. ISBN 0-89791-962-9. Citado na página 39.

JAHRER, M.; TÖSCHER, A. Collaborative filtering ensemble. In: *Proceedings of the 2011 International Conference on KDD Cup 2011 - Volume 18*. [S.l.]: ACM, 2011. p. 61–74. Citado na página 38.

JANNACH, D.; ZANKER, M.; FELFERNIG, A.; FRIEDRICH, G. *Recommender systems: an introduction*. [S.l.: s.n.], 2010. Citado 3 vezes nas páginas 18, 32 e 36.

JOHNSON, W. B.; LINDENSTRAUSS, J.; SCHECHTMAN, G. Extensions of Lipschitz maps into Banach spaces. *Israel Journal of Mathematics*, Springer-Verlag, v. 54, n. 2, p. 129–138, jun 1986. ISSN 0021-2172. Citado na página 46.

KASHNITSKY, Y.; IGNATOV, D. I. Can FCA-based recommender system suggest a proper classifier? In: KUZNETSOV, S. O.; NAPOLI, A.; RUDOLPH, S. (Ed.). *CEUR Workshop Proceedings*. [S.l.: s.n.], 2014. (Proceedings of the 3rd International Workshop "What can FCA do for Artificial Intelligence"? (FCA4AI 2014), v. 1257). Citado na página 35.

KOREN, Y. *The BellKor solution to the Netflix Grand Prize*. [S.l.], 2009. 9 p. Citado na página 38.

KOREN, Y.; BELL, R. Advances in collaborative filtering. In: *Recommender Systems Handbook*. [S.l.: s.n.], 2011. p. 145–186. Citado 4 vezes nas páginas 18, 33, 36 e 37.

KOREN, Y.; BELL, R.; VOLINSKY, C. Matrix factorization techniques for recommender systems. *Computer*, v. 42, n. 8, p. 30–37, 2009. Citado 2 vezes nas páginas 33 e 37.

LANCZOS, C. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. [S.l.]: United States Governm. Press Office, 1950. Citado na página 43.

LESKOVEC, J.; RAJARAMAN, A.; ULLMAN, J. D. *Mining of massive datasets*. [S.l.]: Cambridge University Press, 2014. 513 p. Citado 3 vezes nas páginas 40, 44 e 47.

LI, P.; HASTIE, T. J.; CHURCH, K. W. Very sparse random projections. In: ACM. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, New York, USA: ACM Press, 2006. p. 287–296. ISBN 1595933395. Citado 3 vezes nas páginas 46, 55 e 61.

LIN, C.; LU, S.; LIANG, X.; HUA, J.; MUZIK, O. Cocluster analysis of thalamo-cortical fibre tracts extracted from diffusion tensor MRI. *International Journal of Data Mining and Bioinformatics*, v. 2, n. 4, p. 342–361, 2008. ISSN 17485673. Citado na página 31.

LIN, C.; LU, S.; WU, D.; HUA, J.; MUZIK, O. Coclustering based parcellation of human brain cortex using diffusion tensor MRI. In: MANDOIU, I.; ZELIKOVSKY, A. (Ed.). *Bioinformatics Research and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, (Lecture Notes in Computer Science, v. 4463). p. 539–550. ISBN 3540720308; 9783540720300. Citado na página 31.

LIN, C.; PAI, D.; LU, S.; MUZIK, O.; HUA, J. Coclustering for cross-subject fiber tract analysis through diffusion tensor imaging. *IEEE Transactions on Information Technology in Biomedicine*, v. 14, n. 2, p. 514–525, mar 2010. ISSN 10897771. Citado na página 30.

LINDQUIST, M. A.; XU, Y.; NEBEL, M. B.; CAFFO, B. S. Evaluating dynamic bivariate correlations in resting-state fMRI: A comparison study and a new approach. *NeuroImage*, v. 101, p. 531–546, nov 2014. ISSN 10959572. Citado na página 52.

LIU, T.; MOORE, A. W.; GRAY, A.; YANG, K. An investigation of practical approximate nearest neighbor algorithms. In: *Advances in neural information processing systems*. [S.l.]: MIT Press, 2004. p. 825–832. Citado na página 39.

LU, Y.; CARIN, L.; COIFMAN, R.; SHAIN, W.; ROYSAM, B. Quantitative arbor analytics: Unsupervised harmonic co-clustering of populations of brain cell arbors based on L-measure. *Neuroinformatics*, Humana Press Inc., v. 13, n. 1, p. 47–63, 2014. ISSN 15392791. Citado na página 30.

LU, Y.; TRETT, K.; SHAIN, W.; CARIN, L.; COIFMAN, R.; ROYSAM, B. Quantitative profiling of microglia populations using harmonic co-clustering of arbor morphology measurements. In: *Proceedings - International Symposium on Biomedical Imaging*. San Francisco, CA: [s.n.], 2013. p. 1360–1363. ISBN 9781467364546. ISSN 19457928. Citado na página 30.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 1967. p. 281–297. Citado na página 21.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, v. 1, n. 1, p. 24–45, jan 2004. ISSN 1545-5963. Citado 6 vezes nas páginas , 17, 21, 22, 23 e 24.

MADEIRA, S. C.; TEIXEIRA, M. C.; SA-CORREIA, I.; OLIVEIRA, A. L. Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, IEEE Computer Society Press, Los Alamitos, CA, USA, v. 7, n. 1, p. 153–165, jan 2010. ISSN 1545-5963. Citado 5 vezes nas páginas 17, 27, 30, 72 e 81.

MAUDES, J.; RODRÍGUEZ, J. J.; GARCÍA-OSORIO, C.; PARDO, C. Random projections for SVM ensembles. In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. [S.l.: s.n.], 2010. v. 6097 LNAI, n. PART 2, p. 87–95. ISBN 3642130240. Citado na página 47.

MAUDES, J.; RODRÍGUEZ, J. J.; GARCÍA-OSORIO, C.; PARDO, C. Random projections for linear SVM ensembles. *Applied Intelligence*, v. 34, n. 3, p. 347–359, 2011. ISSN 0924669X. Citado na página 47.

MCLAUGHLIN, M. R.; HERLOCKER, J. L. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In: *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA: ACM, 2004. (SIGIR '04), p. 329–336. ISBN 1-58113-881-4. Citado na página 33.

PARK, Y.; PARK, S.; LEE, S.-g.; JUNG, W. Greedy filtering: A scalable algorithm for k-nearest neighbor graph construction. In: BHOWMICK, S.; DYRESON, C.; JENSEN, C.; LEE, M.; MULIANTARA, A.; THALHEIM, B. (Ed.). *Database Systems for Advanced Applications*. [S.l.]: Springer International Publishing, 2014, (Lecture Notes in Computer Science, v. 8421). p. 327–341. ISBN 978-3-319-05809-2. Citado na página 44.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado 3 vezes nas páginas 44, 55 e 74.

REN, Y.; LI, G.; ZHOU, W. A survey of recommendation techniques based on offline data processing. *Concurrency and Computation: Practice and Experience*, 2014. Citado na página 33.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: *Recommender Systems Handbook*. [S.l.: s.n.], 2011. p. 1–35. Citado 2 vezes nas páginas 17 e 32.

RUBINOV, M.; SPORNS, O. Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage*, v. 52, n. 3, p. 1059–1069, 2010. ISSN 10538119. Citado 2 vezes nas páginas 48 e 49.

SABER, H. B.; ELLOUMI, M. A new study on biclustering tools, biclusters validation and evaluation. *International Journal of Computer Science & Engineering Survey*, 2015. Citado na página 22.

SAITO, T.; OKADA, Y. Bicluster-network method and its application to movie recommendation. In: *Knowledge and Systems Engineering*. [S.l.]: Springer, 2014. p. 147–153. Citado na página 35.

SARWAR, B.; KARYPIS, G.; KONSTAN, J.; RIEDL, J. Item-based collaborative filtering recommendation algorithms. *Proceedings of the tenth international conference on World Wide Web - WWW '01*, ACM Press, New York, New York, USA, p. 285–295, 2001. Citado na página 34.

SCHCLAR, A.; ROKACH, L. Random projection ensemble classifiers. *Lecture Notes in Business Information Processing*, Springer Verlag, v. 24, p. 309–316, 2009. ISSN 18651348. Citado na página 47.

SCHCLAR, A.; TSIKINOVSKY, A.; ROKACH, L.; MEISELS, A.; ANTWARG, L. Ensemble methods for improving the performance of neighborhood-based collaborative filtering. In: ACM. *Proceedings of the third ACM conference on Recommender systems*. [S.l.], 2009. p. 261–264. Citado na página 38.

SHANI, G.; GUNAWARDANA, A. Evaluating recommendation systems. In: *Recommender systems handbook*. [S.l.: s.n.], 2011. p. 257–297. Citado na página 56.

SYMEONIDIS, P.; NANOPOULOS, A.; PAPADOPOULOS, A.; MANOLOPOULOS, Y. Nearest-biclusters collaborative filtering. 2006. Citado na página 34.

SYMEONIDIS, P.; NANOPOULOS, A.; PAPADOPOULOS, A.; MANOLOPOULOS, Y. Nearest-biclusters collaborative filtering with constant values. *Advances in web mining ...*, p. 36–55, 2007. Citado na página 34.

SYMEONIDIS, P.; PANAGIOTIS. Matrix and tensor decomposition in recommender systems. In: *Proceedings of the 10th ACM Conference on Recommender Systems - RecSys '16*. New York, New York, USA: ACM Press, 2016. p. 429–430. ISBN 9781450340359. Citado na página 18.

UKKONEN, E. On-line construction of suffix trees. *Algorithmica*, Springer, v. 14, n. 3, p. 249–260, 1995. Citado na página 27.

VERONEZE, R. *Tratamento de Dados Faltantes Empregando Biclusterização com Imputação Múltipla*. Tese (Doutorado) — Universidade Estadual de Campinas, 2011. Citado 2 vezes nas páginas 33 e 36.

VERONEZE, R.; BANERJEE, A.; VON ZUBEN, F. J. Enumerating all maximal biclusters in numerical datasets. *Information Sciences*, 2016. ISSN 0020-0255. Citado na página 22.

WARD, J. H. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, v. 58, n. 301, p. 236–244, 1963. ISSN 01621459. Citado na página 47.

WU, M. Collaborative filtering via ensembles of matrix factorizations. In: *Proceedings of KDD Cup and Workshop*. [S.l.: s.n.], 2007. v. 2007, p. 43–47. Citado na página 38.

XU, B.; BU, J.; CHEN, C.; CAI, D. An exploration of improving collaborative recommender systems via user-item subgroups. In: *Proceedings of the 21st International Conference on World Wide Web*. New York, NY, USA: ACM, 2012. (WWW '12), p. 21–30. ISBN 978-1-4503-1229-5. Citado na página 35.

YOKOYAMA, N.; OKADA, Y. Item recommendation by query-based biclustering method. In: *Knowledge and Systems Engineering*. [S.l.]: Springer, 2014. p. 155–162. Citado na página 35.

ZHANG, Y.-m.; HUANG, K.; GENG, G.; LIU, C.-l. Fast kNN graph construction with locality sensitive hashing. In: *Machine Learning and Knowledge Discovery in Databases*. [S.l.]: Springer, 2013. p. 660–674. Citado 2 vezes nas páginas 39 e 44.

ZHAO, R.; MAO, K. Semi-random projection for dimensionality reduction and extreme learning machine in high-dimensional space. *Computational Intelligence Magazine, IEEE*, v. 10, n. 3, p. 30–41, 2015. ISSN 1556-603X. Citado na página 47.

ZOU, H.; HASTIE, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, v. 67, n. 2, p. 301–320, 2005. ISSN 13697412. Citado na página 55.