Instituto Superior de
Engenharia de Coimbra

Departamento
de Engenharia Informática e Sistemas

Classificação de Dados Biológicos: Características e Classificadores

Dissertação apresentada para a obtenção do grau de Mestre em Engenharia Informática e Sistemas – Desenvolvimento de Software

Autor
Daniel João Bastos Correia

Orientador
Prof. Doutor Carlos Manuel Jorge da Silva Pereira
Instituto Superior de Engenharia de Coimbra

Coimbra, Dezembro, 2012

“We can't solve problems by using the same kind of thinking we used when we created them.”

Albert Einstein

RESUMO

Reconhecendo a importância que o estudo das proteínas desempenha para a compreensão de inúmeros sistemas biológicos, este trabalho tem por objetivo analisar e explorar a efetividade da utilização de técnicas de *data mining* para classificação de proteínas, aplicadas ao caso de estudo da deteção de peptidases.

A metodologia apresentada e avaliada é baseada em técnicas de *text mining* aplicadas à estrutura primária das proteínas, conjugadas com algoritmos de classificação supervisionada. São apresentados resultados para os algoritmos baseados em máquinas de vetor de suporte, nomeadamente C-SVC, One-Class e LASVM (incremental).

Para o caso de estudo da deteção de peptidases, o algoritmo que apresentou melhores resultados foi o C-SVC. A utilização do algoritmo One-Class apresentou uma diminuição da capacidade de deteção de peptidases relativamente ao C-SVC. Apesar disso, o algoritmo One-Class pode ser uma solução de compromisso quando só são conhecidos exemplos positivos.

Através da utilização do algoritmo incremental LASVM, conseguiram-se resultados muito próximos do C-SVC. Contudo, não foi possível superá-los, mas os resultados obtidos apresentam ganhos significativos ao nível do tempo de treino e da complexidade dos modelos gerados, tornando-se um algoritmo bastante válido para aplicar a problemas que disponham de uma grande quantidade de exemplos de treino.

Além da análise e avaliação dos algoritmos, foi também elaborada uma plataforma web, "Bioink Search", que permite aplicar as metodologias descritas para a deteção de peptidases.

Palavras-chave: Proteínas, Deteção de Peptidases, Text Mining, Support Vector Machines, One-Class, LASVM, Web Platform

ABSTRACT

The study of proteins has an important role for the understanding of innumerous biological systems. The main goal of this study is to analyze and explore the effectiveness of d*ata mining* techniques for classification and detection of peptidases.

The methodology presented is based on *text mining* techniques applied to the primary structure of proteins, also combined with classification algorithms. This study reports the results achieved for three distinct algorithms C-SVC, One-Class and LASVM.

Applying the methodology to the case study of peptidases detection, the algorithm that shows the best results was the C-SVC. Using the One-Class algorithm, the results exhibited a decrease of the ability to detect peptidases. Despite that, the One-Class algorithm can be an alternative solution if only positive samples are available for training.

On the other hand, the incremental algorithm LASVM has achieved a performance close to the C-SVC. Despite the best results were obtained with the C-SVC, this algorithm presents significant gains on training time and can build models with lower complexity. So this algorithm is a valid option, to apply in problems with a large amount of training samples.

Finally, a web platform was developed, the "Bioink Search", which allow researchers to use the presented methodology for peptidase detection.

Keywords: Proteins, Peptidase Detection, Text Mining, Support Vector Machines, One-Class, LASVM, Web Platform

AGRADECIMENTOS

Ao Prof. Doutor Carlos Pereira, pela orientação e acompanhamento facultado ao longo da elaboração de toda a dissertação e por ter-se demonstrado sempre disponível.

Gostaria de agradecer ao Laboratório de Investigação e Inovação Tecnológica do DEIS/ISEC (LIIT) pelas ótimas condições e excelente equipamento que facilitou o desenvolvimento deste trabalho.

Ao projeto BIOINK – PTDC/EIA/71770/2006 pelo acolhimento e pela disponibilidade dos vários membros e ainda, por ter facilitado a aquisição de novas competências no sentido de progredir na minha formação académica.

Por fim, mas não menos importante, gostaria de agradecer à Fundação para a Ciência e Tecnologia (FCT) pelas oportunidades únicas que as bolsas de investigação oferecem para o futuro dos alunos e para a investigação.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS

BO	Binary Occurrence
BOW	Bag of Words
CSS	Cascading Style Sheets
C-SVC	C-Support Vector Classification
DIP	Database of Interacting Proteins
DM	Data Mining
FN	False Negative
FP	False Positive
GA	Genetic Algorithm
HTML	Hypertext Markup Language
LIIT	Laboratório de Investigação e Inovação Tecnológica
NCBI	National Center for Biotechnology Information
OCR	Optical Character Recognition
PPI	Protein-Protein Interaction
PSO	Particle Swarm Optimization
RBF	Radial Basis Function
SVM	Support Vector Machines
TF	Term Frequency

TFIDF	Term Frequency–Inverse Document Frequency
TN	True Negative
TO	Term Occurrence
TP	True Positive
WVTool	Word Vector Tool

CAPÍTULO 1. INTRODUÇÃO

A aquisição de novo conhecimento é uma constante desde os tempos mais remotos. No entanto, o crescimento exponencial de informação poderá constituir uma problemática, se esta não for devidamente catalogada e armazenada, permitindo a sua constante atualização. Assim, a utilização plena do conhecimento depende da sua organização e acessibilidade. Desta forma, torna-se essencial construir sistemas que façam uso dessa vasta quantidade de dados catalogados para classificar dados desconhecidos, de modo a alavancar e otimizar a descoberta de conhecimento promissor.

Todas as formas de conhecimento partilham desta realidade, sendo a vertente explorada neste trabalho a relacionada com dados biológicos, mais concretamente as proteínas e sua estrutura e interações que são cruciais em diversas funcionalidades primárias da vida.

O problema de classificação é uma das tarefas mais comuns de *data mining*, podendo ser aplicado em diversos problemas. Sendo que, através de um conjunto de exemplos divididos num número finito de classes, a classificação é a tarefa automática que determina a classe de um exemplo desconhecido, baseado num modelo previamente treinado com um conjunto de exemplos previamente catalogados.

No âmbito deste trabalho são apresentadas e avaliadas abordagens que permitem classificar proteínas através da sua cadeia de aminoácidos.

As proteínas são compostos orgânicos constituídos por aminoácidos, sendo abundantes nas células e em diversos tecidos. São fulcrais em diversas atividades biológicas, nomeadamente a nível estrutural e dinâmico.

Assim, as funções proteicas incluem a participação no transporte de substâncias, recetores, efeitos imunitários, substâncias hormonais, obtenção de energia, função enzimática, entre outras.

Múltiplas reações bioquímicas do organismo são realizadas com o auxílio de enzimas. As peptidases são um exemplo de enzimas que catalisam a hidrólise das ligações peptídicas

entre os aminoácidos das proteínas. A sua deteção e caracterização são centrais para um maior conhecimento da participação das peptidases nos sistemas biológicos.

Assim, o foco principal da investigação realizada baseia-se no caso de estudo da deteção de peptidases, no qual pretende-se demonstrar a aplicabilidade e eficácia das metodologias de classificação e extração de características. Para o efeito são aplicadas metodologias baseadas em técnicas de *text mining* conjugadas com classificadores supervisionados.

Com o intuito de validar a metodologia usada, efetuou-se uma avaliação preliminar a um outro caso de estudo, a identificação de interações entre proteínas. Neste caso de estudo foram aplicadas as mesmas metodologias.

No decorrer da investigação levada a cabo foi construída uma plataforma específica para o caso de estudo abordado, tendo como objetivo facilitar a deteção de peptidases através da cadeia de aminoácidos de uma proteína.

Ao longo do primeiro capítulo apresentam-se os objetivos traçados para este trabalho, os resultados e contribuições relevantes, os artigos publicados e por último descreve-se a organização da restante tese.

1.1. Objetivos

Dado o rápido crescimento da quantidade de dados biológicos, existe uma crescente demanda por ferramentas computacionais que permitam agilizar o processo de classificação e análise de novos dados. Assim, os principais objetivos propostos para esta dissertação vão ao encontro dessas necessidades, sendo os seguintes:

1 Estudo e avaliação de algoritmos de *machine learning* eficientes, do ponto de vista computacional, para aplicação em vários casos de estudo com dados reais;

2 Implementar e avaliar métodos para o processamento/reconhecimento de sequências de proteínas recorrendo a técnicas de *text mining* para extração de características;

3 Aplicação e avaliação das metodologias ao caso de estudo da deteção de peptidases;

4 Desenvolvimento de uma ferramenta computacional para deteção de peptidases.

1.2. Resultados e contribuições relevantes

A concretização desta investigação foi conseguida através da aplicação de metodologias de *text mining*, conjugadas com classificadores supervisionados. Esta metodologia foi aplicada ao caso de estudo da deteção de peptidases e posteriormente validada com o caso de estudo da identificação de interações entre proteínas.

Assim, como resultado da investigação desenvolvida, esta dissertação, apresenta as seguintes contribuições:

1. Avaliação da utilização de técnicas de *text mining* para a extração de características de sequências de proteínas;

2. Análise comparativa de vários algoritmos baseados em máquinas de vetor de suporte, nomeadamente C-SVC, One-Class e LASVM (incremental), para classificação de sequências de proteínas, mais especificamente a Deteção de peptidases;

3. Disponibilização de uma plataforma web (Bioink Search[1]) que permite a deteção de peptidases a partir da sequência de aminoácidos de uma proteína.

Nesta área foram ainda elaborados alguns artigos científicos que serão enumerados na próxima secção deste capítulo.

[1] http://www.bioink.org/bioinksearch

1.3. Lista de Publicações

Parte do trabalho desenvolvido no âmbito desta dissertação está incluído nas publicações que são enumeradas seguidamente.

I. **D. Correia**, C. Pereira, P. Veríssimo, A. Dourado. *A Platform for Peptidase Detection based on Text Mining Techniques*, Computational Intelligence and Decision Making Intelligent Systems, Control and Automation: Science and Engineering Volume 61, 2013, pp 449-459;

II. N. Lopes, **D. Correia**, C. Pereira, B. Ribeiro e A. Dourado. *An Incremental Hypersphere Learning Framework for Protein Membership Prediction*, International Conference on Hybrid Artificial Intelligence Systems, Salamanca, Spain (2012);

III. **D. Correia**, C. Pereira, P. Veríssimo, A. Dourado. *A Platform for Peptidase Detection based on Text Mining Techniques*, International Symposium on Computational Intelligence for Engineering Systems, Coimbra, Portugal (2011);

IV. C. Pereira, L. Morgado, **D. Correia**, P. Veríssimo, A. Dourado. *Kernel Machines for Proteomics Data Analysis: Algorithms and Tools*, European Network for Business and Industrial Statistics, Coimbra, Portugal (2011);

1.4. Organização e temas abordados

Para além deste capítulo introdutório, a presente dissertação está organizada em mais 6 capítulos, sendo o conteúdo de cada um deles esboçado da seguinte forma:

- Capítulo 2 – Abordagem de Conceitos, local onde são explorados os conceitos chave da dissertação e do caso de estudo abordado;

- Capítulo 3 – Metodologia, introduz-se a metodologia utilizada para procedermos à classificação de dados biológicos recorrendo a técnicas de *text mining*;

- Capítulo 4 – Tecnologias e Ferramentas, apresentam-se as tecnologias utilizadas para elaboração das plataformas web;

- Capítulo 5 – Caso de Estudo: Deteção de Peptidases, expõe-se a abordagem usada, o *dataset* e os resultados obtidos com os vários algoritmos (C-SVC, One-Class e LASVM);

- Capítulo 6 – Plataforma Bioink Search, apresenta-se a plataforma web desenvolvida para deteção de peptidases recorrendo à metodologia apresentada;

- Capítulo 7 – Conclusões e Perspetivas Futuras, expõem-se as conclusões retiradas com a investigação efetuada ao longo desta dissertação. Sugerem-se ainda algumas abordagens possíveis para investigação futura, de forma a dar continuidade ao trabalho desenvolvido.

CAPÍTULO 2. ABORDAGEM DE CONCEITOS

Os possíveis domínios de aplicação da classificação através de algoritmos de *Machine Learning* são vastos. A rápida expansão da quantidade dos dados biológicos tem aumentado a necessidade da criação de métodos e ferramentas que permitam classificar novos dados.

Nesta dissertação restringiu-se o âmbito apenas ao domínio das sequências de proteínas, onde abordamos o caso de estudo deteção de peptidases, e na fase final validamos a metodologia com um outro caso de estudo que se prende na identificação de interações entre proteínas.

O objetivo deste capítulo é expor os principais conceitos dos casos de estudo abordados neste trabalho. Assim, inicia-se pela apresentação do principal foco deste trabalho que são as Proteínas e principais conceitos relacionados com as mesmas. Seguidamente são apresentadas as Peptidases, a utilidade da classificação de proteínas e termina-se o capítulo expondo de uma forma sucinta o que é e qual a importância da identificação de interações entre proteínas.

2.1. Proteínas

As proteínas são biopolímeros organizados com estrutura tridimensional e elevado peso molecular. Compostos abundantes nas células e nos diversos tecidos, resultam da polimerização de 20 tipos de aminoácidos essenciais. Assim, os aminoácidos são identificados como unidades estruturais das proteínas (monómeros), que por sua vez são compostas por elementos como o carbono (C), hidrogénio (H), oxigénio (O), azoto (N) e por vezes enxofre (S). Todos os aminoácidos possuem um átomo de carbono α (C_α) ligado covalentemente a um grupo amina (-NH_2), um grupo carboxilo (-COOH) e um átomo de hidrogénio (H). A cadeia lateral juntamente com o tamanho, forma, carga e reatividade são características que permitem diferenciar os diferentes tipos de aminoácidos (Halpern 1997).

A ligação entre os diferentes aminoácidos dá origem a uma estrutura linear, específica e de comprimento preciso, determinada geneticamente. Esta ligação é realizada através da junção

de um grupo carboxilo com um grupo amina do seguinte aminoácido e consequente perda de uma molécula de água.

Estas ligações permitem o estabelecimento de cadeias polipeptídicas que por sua vez originam macromoléculas como as proteínas, que no mínimo são constituídas por 20 aminoácidos.

As proteínas podem ser classificadas quanto à sua estrutura como primária, secundária, terciária e quaternária. Esta divisão é baseada em algumas características da cadeia proteica como o tipo de aminoácidos, tamanho e configuração.

A estrutura mais simplista baseia-se numa sequência linear de aminoácidos, sendo por isto designada por estrutura primária. Estrutura esta que será o foco central de toda a pesquisa realizada neste trabalho.

Já a estrutura secundária resulta do rearranjo espacial dos resíduos que se encontram próximo da sequência linear. As configurações espaciais mais comuns que a proteína pode adquirir são a hélice-alfa e a folha-beta.

A estrutura terciária resulta de um arranjo tridimensional de uma estrutura secundária.

Por último, a estrutura quaternária assenta em proteínas com duas ou mais cadeias polipeptídicas (subunidades). Estas só se tornam biologicamente ativas quando resulta da associação das várias subunidades com a estrutura terciária.

As funções das proteínas são amplas contribuindo tanto a nível estrutural como dinâmico. Exemplos de proteínas estruturais são o colagénio, a queratina e a elastina. Encontram-se abundantemente em tecidos e são fundamentais no suporte, aquisição de apetências elásticas e protetoras. Biologicamente ativas podem participar no transporte de substâncias, recetores, efeitos imunitários, substâncias hormonais, obtenção de energia, função enzimática, entre outras.

A função enzimática é vital em múltiplas reações bioquímicas do organismo. As enzimas catalisam as reações bioquímicas, sendo específicas para o efeito e reutilizadas no final desta. Um exemplo de enzimas são as peptidases, foco central neste trabalho.

Assim, as peptidases são enzimas que catalisam reações enzimáticas em que há decomposição das proteínas em moléculas de menor tamanho. A sua deteção e

caracterização são centrais para um maior conhecimento da participação das peptidases nos sistemas biológicos

2.2. Peptidases

Designadas também por proteinases, proteases ou enzimas proteolíticas, as peptidases são enzimas responsáveis pela hidrólise das ligações peptídicas entre os aminoácidos que constituem uma proteína. Através destas reações enzimáticas é possível romper as pontes estabelecidas entre proteínas e obter moléculas de menor dimensão.

Consoante o processo de clivagem ocorra a nível das ligações peptídicas internas ou externas, as peptidases podem ser divididas em endopeptidases e exopeptidases respetivamente. Assim, nas endopeptidases a clivagem ocorre na região interna da cadeia polipeptídica entre as regiões N- e C- terminal, sendo negativamente influenciada pela presença de grupos α-amino ou α-carboxilo a nível da atividade enzimática. Por outro lado, nas exopeptidases a quebra de ligações peptídicas sucede apenas nas regiões N- ou C-terminais das cadeias polipeptídicas. De salientar que as peptidases que atuam na região amino terminal livre leva à libertação de um único resíduo de aminoácido (aminopeptidases), um dipeptídeo (dipeptidilpeptidases) ou um tripeptídeo (tripeptidilpeptidases). No entanto, as exopeptidases que atuam na região carboxilo terminal livre produzem um único aminoácido (carboxipeptidases) ou um dipeptídeo (peptidildipeptidases) (Beynon e Bond 1989).

Para além disto, as peptidases ou proteases podem ser classificadas em 7 classes, isto é, proteases de serina, treonina, asparagina, cisteína, ácido glutâmico, ácido aspártico e metaloproteases.

As peptidases envolvidas neste processo de clivagem proteolítica são de vital importância em atividades biológicas essenciais como a modulação da atividade hormonal, dos fatores de coagulação, digestão e absorção de nutrientes, ativação do sistema imune, diferenciação e destruição celular.

A proteólise induzida pelas peptidases pode ser limitada quando há quebra de ligações peptídicas numa sequência específica de aminoácidos ou, por outro lado, pode ser ilimitada

quando leva à degradação do peptídeo na sua íntegra como ocorre na digestão das proteínas e subsequente absorção de aminoácidos, seus monómeros.

De realçar que a clivagem específica de uma proteína pode conduzir à neutralização da peptidase envolvida neste processo ou permitir que esta assuma uma conformação ativa e assim, desempenhar a sua função. Por outro lado, a função das peptidases pode também ser inibida por enzimas inibidoras de protease.

Com isto se depreende que uma alteração na sua constituição ou um desequilíbrio funcional conduz a um estado patológico que pode ser alvo terapêutico. Portanto, a sua deteção e caracterização são centrais para um maior conhecimento da participação das peptidases nos sistemas biológicos, sendo um alvo importante na pesquisa médica e biotecnológica (Hooper 2002).

O crescimento exponencial de informação e a constante atualização da informação pré-existente levou à necessidade de criação de uma base de dados. MEROPS (Rawlings, Barrett et al. 2012) é uma base de dados que foi desenvolvida com o intuito de conter toda a informação relativa à classificação e nomenclatura das peptidases e dos seus inibidores, permitindo a sua consulta e constante atualização.

2.3. Classificação de Proteínas

Nos últimos anos, uma grande quantidade de sequências de DNA e de proteínas têm sido disponibilizados em bases de dados para utilização pública, tais como o GeneBank (Fau, Karsch-Mizrachi et al. 2011), UniProt (Consortium 2012) e a EMBL Nucleotide Sequence Database (Cochrane, Akhtar et al. 2009).

Devido a este crescente aumento de dados disponibilizados emergiu a aplicação de abordagem de *data mining* a esses dados, de forma a proceder à descoberta de conhecimento, tal como a classificação de proteínas, permitindo assim auxiliar na descoberta da função de novas proteínas.

Contudo, existem amplas formas de classificar as proteínas. O objetivo major da classificação é uma mais fácil organização e inclusão de informação já adquirida, assim como de novos conhecimentos. As principais classificações de proteínas assentam essencialmente na sua estrutura, função e tamanho.

Este trabalho foca-se essencialmente na deteção de peptidases, supramencionadas como enzimas que se incluem num subtipo de função das proteínas (função enzimática).

No entanto, a abordagem apresentada neste trabalho poderá ser extensível a outros tipos de classificações.

2.4. Interação entre Proteínas

A maioria dos processos biológicos são desempenhados com o auxílio de uma ou mais proteínas. Desde o crescimento celular, proliferação, obtenção de nutrientes, expressão de genes, funções de motilidade, comunicação intercelular e, por fim, apoptose ou seja morte celular. As mais diversas funções celulares dependem da interação proteína-proteína que pressupõe a participação de duas ou mais proteínas em conjunto (De Las Rivas e Fontanillo 2010).

Inevitavelmente para uma melhor compreensão da participação das proteínas nestas tarefas, é primordial o estudo, não só da proteína como molécula isolada, mas também da sua interligação com outras proteínas envolvidas no processo.

Para tal, as interações entre proteínas são estudadas do ponto de vista bioquímico, dinâmica molecular, sinal, transdução e conexões metabólicas ou genéticas, entre outros (Thermo Scientific 2010).

O conhecimento adquirido neste âmbito é imprescindível para o desenvolvimento de novas abordagens científicas que permitirão o desenvolvimento de técnicas diagnósticas e terapêuticas revolucionárias, nomeadamente na área oncológica.

CAPÍTULO 3. METODOLOGIA

Neste trabalho optou-se pela escolha da metodologia de reconhecimento de padrões tradicional (apresentada na Figura 1). Para o efeito, primeiramente é necessário efetuar a extração de características, de maneira a transformar os dados que se pretendem classificar numa estrutura organizada, tendo-se optado pela utilização do vetor de características também designado por BOW (bag of words).

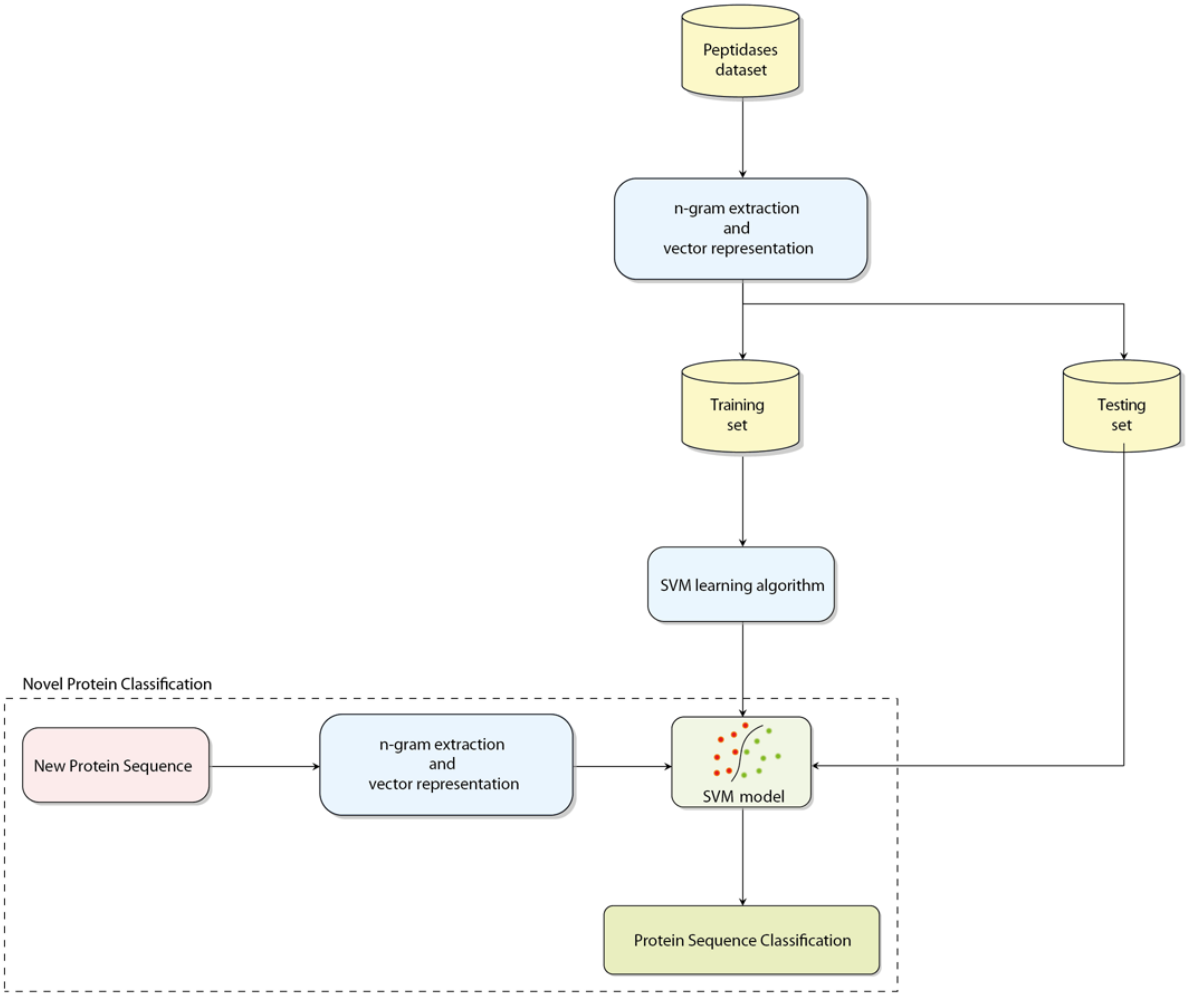


Figura 1 – Abordagem usada para classificação de proteínas.

Posteriormente, utilizando um algoritmo de aprendizagem supervisionada é efetuado o treino de um modelo com o vetor de características extraídas na primeira.

O modelo gerado na fase de treino é então utilizado, sempre que seja necessário classificar novos dados, uma vez que esse modelo idealmente conseguirá ter uma representação que descriminará de forma inequívoca as várias classes para o qual foi treinado.

Este trabalho segue algumas das metodologias apresentadas por Cheng, Carbonell et al. (2005), Tomović, Janičić et al. (2006) e ainda Caridade (2010), uma vez que estes artigos reportam a utilização de *n*-grams para classificação e clustering de proteínas e sequências do genoma.

Contudo, neste trabalho é abordado um caso de estudo distinto e mais específico, a deteção de peptidases. Para o efeito aplicaram-se técnicas de *text mining* e foram utilizados diferentes algoritmos que posteriormente foram avaliados e comparados entre si.

Ao longo desta secção são apresentados, mais pormenorizadamente, cada um dos processos desta abordagem, a extração de características (ver 3.1 – Extração de Características), classificação (ver secção 3.3 – Algoritmos de Classificação Supervisionada) e posteriormente apresentam-se as métricas usadas para avaliar os resultados (secção 0 – Métricas de Avaliação). Por fim, apresenta-se o método usado para seleção de parâmetros (secção 3.5 – Seleção de Parâmetros).

3.1. Extração de Características

Independentemente do algoritmo e do tipo de dados que se pretendem classificar, o processo de classificação inicia-se sempre por uma fase preliminar, a extração de características. Esta fase tem como objetivo primordial a obtenção dos dados mais relevantes para a distinção, e ainda construir uma representação dos mesmos de forma estruturada. Só através de uma representação simples e sólida é possível utilizar esses dados em algoritmos de classificação. Na secção 3.2 – Representação das Características, será abordada a forma de representação usada neste trabalho.

Esta extração é de extrema importância, já que o desempenho dos algoritmos de classificação degrada-se com a seleção de características irrelevantes e ainda, aumenta a complexidade do

mesmo com o aumento do número de características. Assim, idealmente pretende-se obter o menor número de características que consigam ser o mais discriminantes possíveis para as classes e tipo de dados que se pretende classificar.

Como o principal foco deste trabalho passa pela aplicação de técnicas de *text mining* na extração de características da estrutura primária das proteínas, utilizaram-se essencialmente dados no formato FASTA[2]. Este formato define uma forma simples para representar sequências de proteínas, sendo constituído por um cabeçalho, com uma descrição da proteína, e uma linha com a sequência de aminoácidos propriamente dita por cada proteína (ver Figura 2).

Figura 2 – Sequências de Proteínas no formato FASTA.

Assim, foi necessário efetuar a extração de características a ficheiros com este formato. Para tal, utilizou-se a biblioteca WVTool (Wurst 2007) apresentada em mais detalhe na secção 4.1 – WVTool.

[2] Especificação do formato FASTA no NCBI – http://blast.ncbi.nlm.nih.gov/blastcgihelp.shtml

Como neste projeto utiliza-se essencialmente tecnologia ASP.NET e esta biblioteca foi desenvolvida em Java, foi necessário poder incorporá-la numa aplicação desenvolvida em ASP.NET. Para isso, usou-se o IKVM (Frijters) que permitiu criar uma DLL (Dynamic-Link Library) da biblioteca, facilitando a sua utilização.

A metodologia para extração de características aplicada neste trabalho é baseada em técnicas de *text mining*. Fundamentalmente, a sequência de aminoácidos que compõem uma proteína, será subdividida em pequenos trechos de aminoácidos, não alterando a sua ordem. Esses pequenos trechos são os *n*-grams. Na Figura 3 é apresentada a forma como é feita essa divisão.

Figura 3 – Extração e representação dos *n*-grams de uma sequência de aminoácidos.

Definindo o conceito de *n*-grams, dada uma sequências de caracteres $S = (s_1, s_2, \dots, s_{N+(n-1)})$, onde N e n são número positivos inteiros, um *n*-gram da sequência de caracteres S é um subsequente e consecutivo conjunto de caracteres de tamanho n (Tomović, Janičić et al. 2006).

Generalizando, um qualquer *n*-gram é dado pela sequência $(s_i, s_{i+1}, \dots, s_{N+(n-1)})$, onde *n* é o tamanho do *n*-gram. Seguidamente na Figura 4 é apresentado um exemplo prático da criação de *n*-grams de 2.

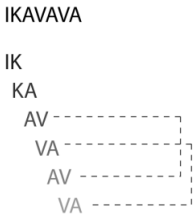


Figura 4 – Exemplo detalhado dos *n*-grams de 2 (bigrams) extraídos da sequência de aminoácidos.

Os *n*-grams têm sido aplicados a um vasto número de casos de estudo e de domínios diferentes tais como, compressão de texto, correção ortográfica, reconhecimento ótico de caracteres

(OCR), categorização de texto automatizado entre vários outros, tendo apresentado bons resultados com a sua utilização (Tomović, Janičić et al. 2006).

Um dos principais problemas da utilização de *n*-grams passa pela explosão exponencial do número de possíveis combinações. No presente trabalho, esse problema foi solucionado através da aplicação de *prunning*, que permite definir um número mínimo e máximo de ocorrências necessárias em todo o *dataset*, para que uma característica possa ser considerada como válida, sendo que as que não cumprirem esse requisito são eliminadas.

De forma a efetuar-se a extração de características de uma forma simples e rápida, foi elaborada uma aplicação que permite efetuar várias combinações de n-grams e tipos de métricas (ver Figura 5). No Anexo A, é possível visualizar o digrama de classes desta aplicação.

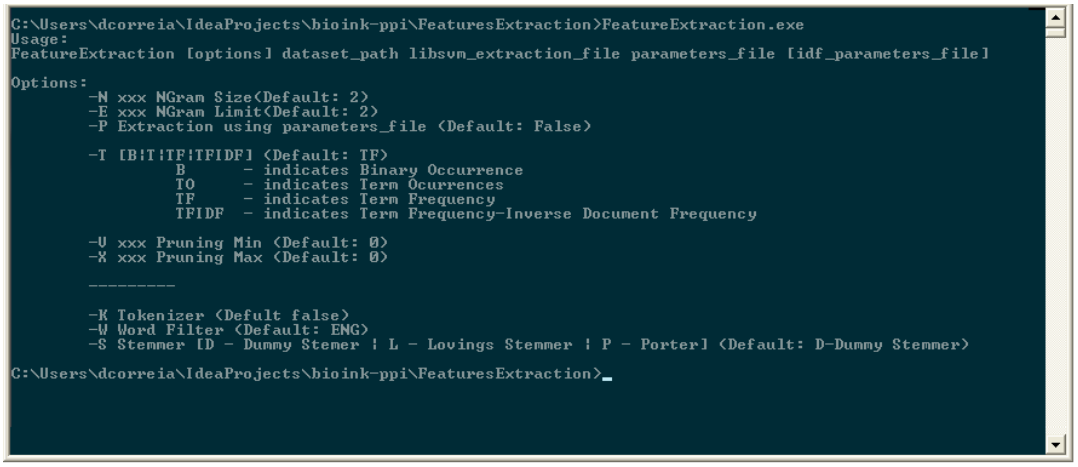


Figura 5 – Aplicação desenvolvida para extração de características.

Através da utilização desta aplicação é possível definir o tamanho do *n*-gram que se pretende usar, o *stemmer*, o tipo de métrica de contabilização das características, os valores máximos e mínimos de *pruning* e ainda definir a localização do *dataset* e dos ficheiros gerados pelo processo de extração.

O *stemmer* consiste num processo que tem por objetivo reduzir uma palavra à sua raiz. Existem diversos tipos de *stemmer*, em que normalmente são específicos para a língua dos documentos, já que a redução à sua raiz depende das especificidades da língua.

Por sua vez, o *pruning* é um critério para o qual é definido um limite mínimo e máximo de exemplos nos quais uma característica deve aparecer para ser considerada válida. Uma vez que tanto uma característica que aparece poucas vezes como uma que apareça muitas vezes, poderão não ter grande capacidade discriminativa.

3.2. Representação das Características

Independentemente das características extraídas é sempre necessário representá-las numa estrutura organizada. Para o efeito, neste trabalho foi utilizado o vetor de características também conhecido por Bag of words (BOW) inicialmente introduzido por Salton, Wong et al. (1975).

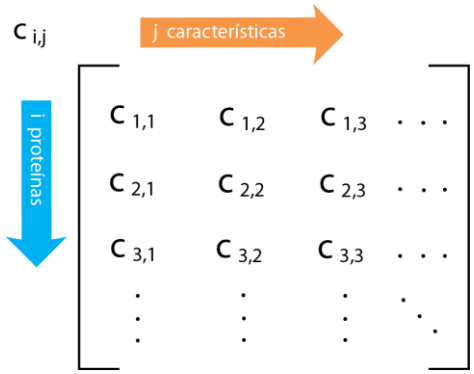


Figura 6 – Estrutura de um vetor de caraterísticas.

Este vetor de características é constituído por um vetor de duas dimensões, onde cada linha corresponde a um exemplo (sequência de um proteína, documento, imagem, etc.) e cada coluna representa uma característica. Na Figura 7 é possível observar um vetor de características de uma extração a sequências de proteínas. O valor atribuído a cada uma das células representa a relevância da característica, sendo que esse valor dependerá da função de cálculo usada. Na secção 3.2.1- Cálculo da Relevância das Características são apresentados os métodos de cálculo usados neste trabalho.

EV	VQ	QL	LQ	QQ	QS	SG	GA	AE
0.0749531688995862	0.149906337799172	0.0749531688995862	0	0	0.0749531688995862	0.149906337799172	0.149906337799172	0.0749531688
0.158113883008419	0	0	0	0.0790569415042095	0.0790569415042095	0.316227766016838	0	0
0	0	0	0	0	0.0368855556781658	0.0368855556781658	0.0737711113563317	0.1106566670
0	0.0619875727373744	0	0.0309937863686872	0	0.123975145474749	0.123975145474749	0.0619875727373744	0.0309937863
0.0689655172413792	0	0	0.0344827586206896	0.0344827586206896	0	0.0344827586206896	0.0689655172413792	0.1034482758
0.0487370178828579	0	0.146211053648574	0.0487370178828579	0.0487370178828579	0	0.0487370178828579	0	0.0487370178
0.0556414884074657	0	0.0834622326111986	0.111282976814931	0	0	0.0834622326111986	0.0556414884074657	0.0278207442
0.0274954679174724	0.0274954679174724	0.10998187166989	0.0549909358349449	0.0274954679174724	0.0137477339587362	0.151225073546098	0.0549909358349449	0.0687386697
0.0749531688995862	0.149906337799172	0.0749531688995862	0	0	0.0749531688995862	0.149906337799172	0.149906337799172	0.0749531688
0.158113883008419	0	0	0	0.0790569415042095	0.0790569415042095	0.316227766016838	0	0
0	0	0	0	0	0.0368855556781658	0.0368855556781658	0.0737711113563317	0.1106566670
0.158113883008419	0	0	0	0.0790569415042095	0.0790569415042095	0.316227766016838	0	0
0	0.0619875727373744	0	0.0309937863686872	0	0.123975145474749	0.123975145474749	0.0619875727373744	0.0309937863
0.0689655172413792	0	0	0.0344827586206896	0.0344827586206896	0	0.0344827586206896	0.0689655172413792	0.1034482758
0.0487370178828579	0	0.146211053648574	0.0487370178828579	0.0487370178828579	0	0.0487370178828579	0	0.0487370178
0.0556414884074657	0	0.0834622326111986	0.111282976814931	0	0	0.0834622326111986	0.0556414884074657	0.0278207442
0.0274954679174724	0.0274954679174724	0.10998187166989	0.0549909358349449	0.0274954679174724	0.0137477339587362	0.151225073546098	0.0549909358349449	0.0687386697
0.0749531688995862	0.149906337799172	0.0749531688995862	0	0	0.0749531688995862	0.149906337799172	0.149906337799172	0.0749531688
0.158113883008419	0	0	0	0.0790569415042095	0.0790569415042095	0.316227766016838	0	0
0	0	0	0	0	0.0368855556781658	0.0368855556781658	0.0737711113563317	0.1106566670
0	0.0619875727373744	0	0.0309937863686872	0	0.123975145474749	0.123975145474749	0.0619875727373744	0.0309937863

Figura 7 – Exemplo de um vetor de características parcial de uma extração para *n*-grams de 2.

3.2.1. Cálculo da Relevância das Características

Após as características serem extraídas, é necessário utilizar uma métrica de contabilização da relevância das características. Existem diversos métodos para o fazer, sendo que no presente trabalho utilizaram-se as seguintes, *Binary Occurrence (BO) (ver equação* – 2.1.1*), Term Occurrence (TO) (*ver *equação* – 2.1.2*), Term Frequency (TF) (ver equação* – 2.1.3*)* e *Term frequency – Inverse Document Frequency (TF-IDF) (ver equação* – 2.1.4*).*

Essas métricas são obtidas tendo em conta as três contagens apresentadas abaixo:

f_{ij} – Número de ocorrências da característica *i* no exemplo *j*;

fd_j – Número total de termos no exemplo *j*;

ft_i – Número total de documentos onde a característica *i* aparece;

D – Número total de documentos.

A ocorrência binária (*Binary Occurrence*) é dada pela equação 3.1.1.

$$v_{ij} = \begin{cases} 1, & f_{ij} > 0 \\ 0, & f_{ij} \leq 0 \end{cases} \quad \textbf{(3.2.1)}$$

O número de ocorrências de uma característica (*Term Occurrence*) é dado pelo número absoluto de ocorrências da característica num dado exemplo (ver equação 3.1.2).

$$TO_{ij} = f_{ij} \quad \textbf{(3.2.2)}$$

A frequência de um termo (*Term Frequency*) é dada pela equação 3.1.3.

$$TF_{ij} = \frac{f_{ij}}{fd_j} \quad \textbf{(3.2.3)}$$

O TFIDF é dado pela equação 3.1.4, salientando-se que esta métrica tende a diminuir a contagem das características mais comuns.

$$idf_i = \log\left(\frac{|D|}{ft_i}\right)$$

$$TFIDF_{ij} = tf_{ij} \times idf_i \quad \textbf{(3.2.4)}$$

3.3. Algoritmos de Classificação Supervisionada

A classificação supervisionada depende sempre da existência de amostras de treino que sejam representativas das classes a que os dados pertencem e o mais uniforme possível.

Nesta secção procede-se à apresentação dos algoritmos de classificação supervisionada usados neste trabalho.

3.3.1. Support Vector Machines

Support Vector Machines (Cortes e Vapnik 1995) representa um conjunto de técnicas e métodos de aprendizagem supervisionada usados para fins de classificação e regressão, tendo por base a Teoria de Aprendizagem Estatística de Vapnik (Vapnik 1995; Vapnik 1998).

O algoritmo com base em dados previamente classificados, permite que seja efetuado um treino do qual resulta um modelo. Posteriormente, este modelo é aplicado a novos conjuntos de dados desconhecidos, com o intuito de os classificar corretamente.

Durante o processo de treino, o algoritmo tem como objetivo encontrar a melhor divisão do espaço das características por hiperplanos, de forma a conseguir a melhor separação das várias classes existentes. Na Figura 8 é apresentado um pequeno exemplo dessa separação.

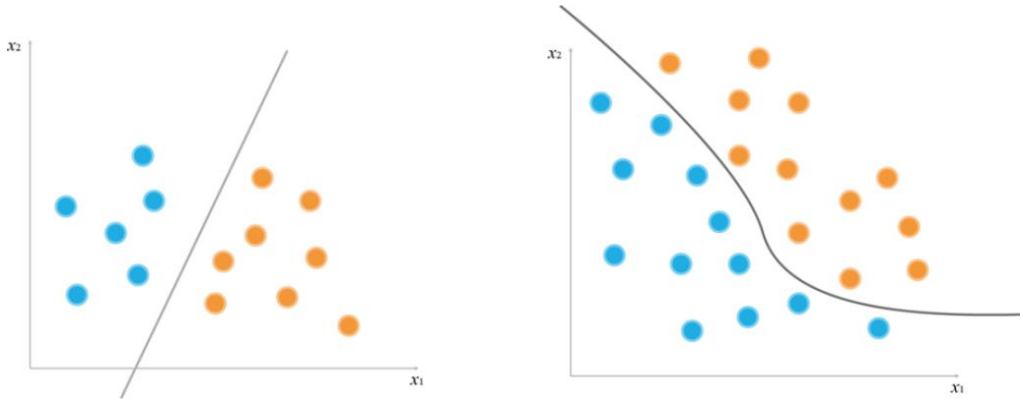


Figura 8 – Exemplo da separação de duas classes com apenas duas características.

A delineação dos hiperplanos é feita através dos vetores, sendo estes construídos a partir de algumas das características usadas na fase de treino. A escolha das características é feita de modo a que seja possível construir hiperplanos que permitam a melhor separação possível, como pode ser observado na Figura 9.

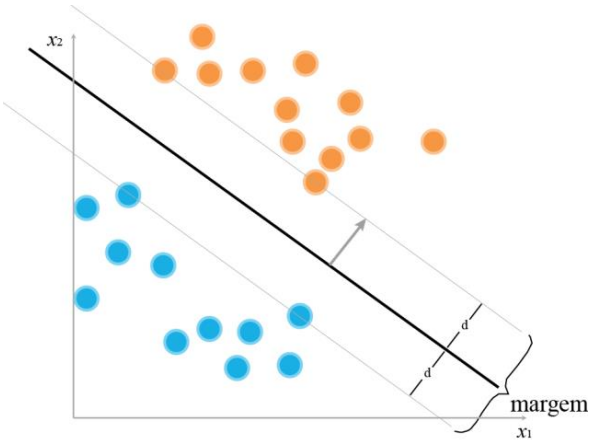


Figura 9 – Hiperplano de separação ideal com uma margem máxima.

O hiperplano que possibilita a melhor separação possível é aquele que consiga delimitar as fronteiras das classes envolvidas com a maior distância entre as classes envolvidas. A utilização da margem máxima possibilita a criação de modelos que sejam generalizados e não fiquem tão sujeitos ao problema de *overfitting* sobre os dados de treino.

O classificador C-SVC resolve o seguinte problema de otimização (Chang e Lin 2011):

$$\min_{w,b,\varepsilon} \quad \frac{1}{2} w^t w + C \sum_{i=1}^{l} \varepsilon_i \qquad \textbf{(3.3.1)}$$

Sujeito a:

$$y_i(w^t \emptyset(x_i) + b) \geq 1 - \varepsilon_i \qquad \textbf{(3.3.2)}$$

$$\min_{w,b,\varepsilon} \quad \frac{1}{2} w^t w + C \sum_{i=1}^{l} \varepsilon_i \qquad \textbf{(3.3.3)}$$

$$\varepsilon_i \geq 0, i = 1, \ldots, l \qquad \textbf{(3.3.4)}$$

A função de decisão é dada por:

$$sgn(w^t \emptyset(x) + b) = sgn(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b)) \quad \textbf{(3.3.5)}$$

Os problemas de classificação não são sempre linearmente separáveis, assim, é necessário efetuar transformações de forma a possibilitar a separação linear do conjunto de dados. Para isso são usados métodos de kernel, que permitem estender espaço hiperdimensional por forma a possibilitar a separação linear dos dados.

Os métodos de kernel mais utilizados são os seguintes:

- Linear:

$$K(x, x') = x \,.\, x' \quad \textbf{(3.3.6)}$$

- Polinomial:

$$K(x, x') = (x^T x' + 1)^p \quad \textbf{(3.3.7)}$$

- Radial Basis Function (RBF):

$$y_i(w^t \emptyset(x_i) + b) \geq 1 - \varepsilon_i \quad \textbf{(3.3.8)}$$

Durante a elaboração deste trabalho foi também usado o *Cross-Validation* que é um método que consiste em separar um conjunto inicial de dados em *N* subconjuntos, sendo que *N* também é o número de iterações a efetuar. Em cada iteração são utilizados N-1 conjuntos para treino e o conjunto excluído para teste. O processo é executado N vezes permutando de forma circular os conjuntos. No final, o modelo devolvido é o resultante do conjunto de treino que obteve a melhor validação.

Existem um conjunto diversificado de implementações deste algoritmo, contudo neste trabalho a biblioteca usada foi LIBSVM versão 3.1[3] (Chang e Lin 2011). Optou-se pela utilização deste algoritmo por ser amplamente conhecido e testado, permitindo ser uma ótima base para avaliação da abordagem apresentada e ainda servirá para comparar com outros algoritmos. Além disso, este algoritmo também foi classificado em terceiro lugar na IEEE International Conference on Data Mining (Wu e Kumar 2009).

[3] Disponível em http://www.csie.ntu.edu.tw/~cjlin/libsvm/

3.3.2. One-class SVM

A principal característica de classificadores One-Class SVM é o facto de este conseguir construir um modelo discriminativo utilizando apenas exemplos positivos. Schölkopf, Platt et al. (2001) propuseram uma abordagem que essencialmente consiste em separar os exemplos positivos da origem através de um hiperplano (na Figura 10 é apresentada uma pequena ilustração dessa separação). Assim, quando é necessário prever a classificação de um novo exemplo, é apenas necessário saber de que lado do hiperplano é que ele fica posicionado no espaço das características, para poder determinar a sua classe.

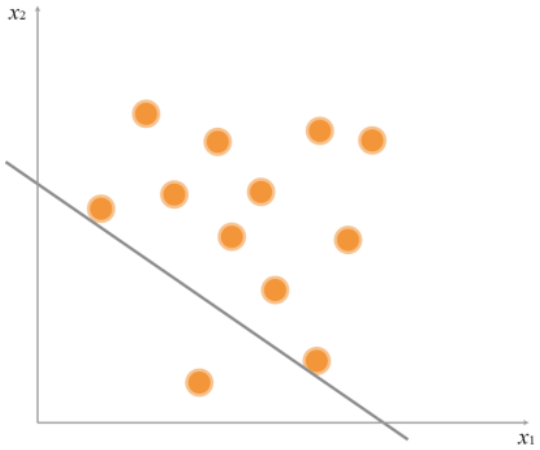


Figura 10 – Exemplo da separação da origem aplicada pelo algoritmo One-Class.

Para determinar o hiperplano o algoritmo One-Class faz o mapeamento dos exemplos de treino para o espaço das características e separa-os da origem com a margem máxima. Os exemplos que não fiquem corretamente separados fazem com que a função objetiva seja penalizada (Schölkopf, Platt et al. 2001).

Para determinar o hiperplano de separação dos dados da origem é resolvida a seguinte função quadrática (Chang e Lin 2011):

$$\min_{w,b,\varepsilon} \quad \frac{1}{2} w^T w - \rho + \frac{1}{vl} \sum_{i=1}^{l} \varepsilon_i \qquad \textbf{(3.3.9)}$$

Sujeito a

$$w^T \emptyset(x_i) \geq \rho - \varepsilon_i \qquad \textbf{(3.3.10)}$$

$$\varepsilon_i \geq 0, i = 1, \ldots, l \quad \textbf{(3.3.11)}$$

Sendo que a superfície de decisão é dada por:

$$sgn(\sum_{i=1}^{l} \alpha_i K(x_i, x) - \rho) \quad \textbf{(3.3.12)}$$

Neste trabalho o algoritmo One-Class usado foi o disponibilizado pela biblioteca LIBSVM versão 3.1.

3.3.3. LASVM

O algoritmo incremental LASVM foi introduzido por Bordes, Ertekin et al. (2005). Este algoritmo é um classificador online (o *dataset* de treino não é todo disponibilizado à partida) que constrói de forma incremental um modelo discriminativo, adicionando ou removendo vetores de suporte de forma iterativa, convergindo para a solução das SVM.

Comparativamente com outros algoritmos de kernel, o LASVM inclui uma fase de remoção de vetores que lhe permite lidar com dados com ruído (Bordes, Ertekin et al. 2005).

Em relação à performance do algoritmo é referido que consegue obter resultados similares aos obtidos com a biblioteca LIBSVM, contudo, consegue esses resultados requerendo menos memória, sendo esse um fator fundamental em termos de velocidade de treino quando se pretende utilizar *datasets* de grandes dimensões.

Além disso, o facto de ser um algoritmo incremental permite treinar um modelo já existente adicionando apenas os novos exemplos que tenham surgido, não sendo necessário treinar o classificador com todos os exemplos existentes, permitindo assim que a incorporação de novos exemplos seja feita de uma forma mais célere.

Neste trabalho utilizou-se o algoritmo LASVM versão 1.1 disponibilizado por Bordes, Ertekin et al. (2005) [4].

[4] Disponível em http://leon.bottou.org/projects/lasvm

3.4. Métricas de Avaliação

Para medir a eficácia e permitir a realização de comparações é necessário utilizar métricas que permitam efetuar uma correta avaliação dos resultados.

Como são essencialmente avaliados classificadores binários, os possíveis resultados obtidos são sempre quatro, podendo ser expressos numa matriz denominada por matriz confusão representada na Figura 11.

		Classe Predita	
		+	-
Actual Class	+	TP	FN
	-	FP	TN

Figura 11 - Matriz Confusão

Onde,

- TP – exemplos positivos corretamente classificados;
- TN – exemplos negativos corretamente classificados;
- FP – exemplos negativos classificados como positivos;
- FN – exemplos positivos classificados como negativos.

Através desta matriz de confusão é então possível calcular um conjunto de métricas que permite uma mais fácil avaliação e comparação dos resultados obtidos entre os vários algoritmos.

Além das comparações, estas métricas permitem ainda criar modelos que possuem melhores resultados numa métrica específica em detrimento de outra. Esse tipo de decisões dependem da área de aplicação.

Accuracy

A *accuracy*, permite calcular a proporção de resultados corretamente classificados.

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad \textbf{(3.4.1)}$$

Sensitivity

A *sensitivity*, permite avaliar a capacidade do classificador identificar exemplos positivos.

$$sensitivity = \frac{TP}{TP + FN} \quad \textbf{(3.4.2)}$$

Specificity

A *specificity*, permite avaliar a capacidade do classificador identificar exemplos negativos.

$$specificity = \frac{TN}{TN + FP} \quad \textbf{(3.4.3)}$$

Quanto maior a especificidade menor será a probabilidade de um exemplo ser classificado como positivo sendo negativo e assim a Taxa de Falsos positivos será baixa.

Precision

Através desta métrica é possível avaliar a precisão de um classificador na identificação de exemplos positivos.

$$precision = \frac{TP}{TP + TN} \quad \textbf{(3.4.4)}$$

F-measure

Esta métrica é essencialmente utilizada no domínio das aplicações de extração de informação (Information retrieval), permitindo obter uma média ponderada entre a *precision* e o *recall.*

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \quad \textbf{(3.4.5)}$$

3.5. Seleção de Parâmetros

A efetividade de um classificador SVM depende da escolha do *kernel*, mas porventura ainda mais importante é a correta escolha dos parâmetros desse *kernel*. Não se sabe de antemão quais são os parâmetros que permitem obter os melhores resultados para um dado problema. O objetivo passa então por escolher os melhores parâmetros, para que o classificador consiga obter uma melhor taxa de acerto na classificação de dados desconhecidos.

Uma das práticas mais comuns de seleção de parâmetros é a pesquisa em grelha, contudo também já foi reportada a utilização de algoritmos genéticos para este efeito, sendo ambos apresentados mais detalhadamente nos próximos subcapítulos.

3.5.1. Pesquisa em Grelha

A pesquisa em grelha é a prática mais comum de seleção de parâmetros e é recomendada por Chih-Wei, Chih-Chung et al. (2003). Este método consiste em usar vários conjuntos de parâmetros para treinar um modelo, sendo posteriormente testados e escolhidos os que obtiveram melhores resultados.

Chih-Wei et al. referem que a melhor forma para seleção dos parâmetros passa pela variação do valor segundo um crescimento exponencial. Concretizando, para efetuar uma pesquisa em grelha para o algoritmo C-SVC com o kernel RBF, que possui dois parâmetros C e γ, poder-se-iam variar esses parâmetros como sugerido por Chih-Wei, Chih-Chung et al. (2003), onde C=$(2^{-5},\ 2^{-4}, \ldots\ \ 2^{15})$ e γ=$(2^{-15},\ 2^{-14}, \ldots\ \ 2^{3})$.

Apesar deste método ser simples e pouco eficiente computacionalmente oferece alguma confiança nos parâmetros escolhidos, pois é feita uma pesquisa exaustiva.

De forma a poder-se utilizar este método de seleção de parâmetros foi elaborada uma aplicação que permite efetuar esta pesquisa (ver Figura 12).

Figura 12 – Utilitário desenvolvido para automatizar o processo de pesquisa em grelha.

Através da utilização desta aplicação é possível definir o algoritmo que se deseja usar, os valores máximos e mínimos dos parâmetros e a localização dos dados a avaliar.

Seguidamente, a aplicação inicia uma pesquisa em grelha efetuando o treino de um modelo por cada combinação de parâmetros. Cada um desses modelos é posteriormente usado para classificar um conjunto de dados de teste, de forma a poder-se obter uma avaliação do modelo gerado pelo classificador com os parâmetros definidos.

Posteriormente, é necessário verificar os resultados obtidos para que se possam escolher os parâmetros que obtiveram melhores cotações.

No Anexo C pode ser visualizado o diagrama de classes desta aplicação.

3.5.2. Algoritmo Genético

Os algoritmos genéticos são métodos de pesquisa probabilística inspirados nos princípios da seleção natural e da genética desenvolvidos por Holland (1975). Inspirado na evolução natural, os algoritmos genéticos iniciam a otimização a partir de uma população inicial (conjunto de indivíduos que são potenciais soluções para o problema) e vão evoluindo ao longo de sucessivas gerações. Isto tem como objetivo encontrar uma solução com a melhor qualidade, em que seria idealmente ótima para o problema em causa.

A cada geração do algoritmo genético é criada uma nova população através da seleção dos indivíduos com a melhor qualidade para o domínio do problema, sendo estes reproduzidos entre si através da aplicação dos operadores genéticos de recombinação e mutação. Este processo evolutivo leva à progressão dos indivíduos, o que permitirá que estes sejam mais aptos do que os seus ancestrais.

A utilização destes algoritmos para seleção de parâmetros e de características para SVMs já foi anteriormente reportado por Huang e Wang (2006). Neste trabalho é seguida uma abordagem semelhante.

Assim, primeiramente será validada apenas a seleção de parâmetros, e posteriormente introduz-se também a seleção de características, que neste caso será a seleção dos *n*-grams presentes.

Esta metodologia será importante, na medida que permitirá automatizar a seleção de parâmetros e ainda selecionar qual a melhor combinação de *n*-grams.

3.5.2.1. Representação

Neste trabalho utilizou-se essencialmente o *kernel* rfb, porque é aquele em que se obtiveram os melhores resultados, e possui apenas dois parâmetros (*C* e o γ).

Assim, é necessário apenas uma representação para esses dois parâmetros. A representação utilizada é bastante comum para este problema, sendo constituída por uma sequência binária com *n* posições. Na Figura 13 é apresentada a composição de um indivíduo baseada na representação apresentada por Huang e Wang (2006).

g_c^1	...	g_c^i	...	$g_c^{n_c}$	g_γ^1	...	g_γ^j	...	$g_\gamma^{n_\gamma}$	g_f^1	...	g_f^k	...	$g_f^{n_f}$

Figura 13 – Composição de um indivíduo dividido em três partes (*C*, γ e características).

Esta representação é dividida em três partes: o parâmetro *C*, γ e as características que neste caso de estudo serão os *n*-gram. Assim, na Figura 13, o valor do parâmetro C é representado por $g^1_c \sim g^{n_c}_c$, o valor do parâmetro γ é representado por $g^1_\gamma \sim g^{n_\gamma}_\gamma$, e por fim, $g^1_f \sim g^{n_f}_f$ identifica se o *n*-gram é selecionado (1 selecionado e o 0 não selecionado). Já n_c e n_y são o número de bits que representaram os parâmetros *C* e γ, respetivamente, e por sua vez o n_f é o número do *n*-gram.

A sequência binária dos parâmetros *C* e γ são transformados no seu valor através da equação 3.4.1, baseada na equação proposta por Huang e Wang (2006).

$$valor\ do\ parâmetro = \min_p + \frac{max_p - min_p}{2^l - 1} \times d \qquad \textbf{(3.4.1)}$$

Onde, $\min_p$ e $\max_p$ são o valor mínimo e máximo respetivamente para o parâmetro, l corresponde ao número de bits utilizados para representar o parâmetro e d é o valor decimal da sequência binária do parâmetro. Na Figura 14 são apresentados três exemplos de possíveis indivíduos, com a respetiva conversão.

Apesar de esta representação ser a mais natural para este problema, existe uma limitação importante que tem de ser levada em conta que é a existência de soluções inválidas no espaço de procura. Essas soluções inválidas ocorrem quando todos os bits que representam os *n*-grams são 0. De forma a lidar com esta situação optou-se por penalizar as soluções inválidas.

C				γ				*n*-gram(1)	*n*-gram(2)	*n*-gram(3)	Conversão
1	1	0	1	0	1	1	0	1	0	1	$C = -5 + \frac{15-(-5)}{2^4-1} \times 13 = 12.33$ $\gamma = 2^{-15+\frac{-15-3}{2^4-1}\times 6} = 2^{-7.8}$ *n*-gram (1, 2)
1	1	1	1	1	1	1	0	0	0	1	$C = -5 + \frac{15-(-5)}{2^4-1} \times 15 = 15$ $\gamma = 2^{-15+\frac{-15-3}{2^4-1}\times 14} = 2^{1.8}$ *n*-gram (3)
1	0	0	1	0	1	0	0	1	1	1	$C = -5 + \frac{15-(-5)}{2^4-1} \times 9 = 7$ $\gamma = 2^{-15+\frac{-15-3}{2^4-1}\times 4} = 2^{10.2}$ *n*-gram (1, 2, 3)

Figura 14 – Exemplos de indivíduos com a respetiva conversão.

Como critério de avaliação usado para determinar a qualidade dos indivíduos optou-se pela utilização do *F-measure*. A utilização desta métrica deveu-se essencialmente ao facto de ser desejável ter uma única métrica que permitisse aferir a performance e comparar os classificadores. Como esta métrica é a média harmónica ponderada entre a precisão e o recall, tornou-se a escolha mais acertada.

3.5.2.2. Implementação

A implementação desta abordagem para seleção de parâmetros e posterior extensão para seleção de características (tipo de *n*-grams) foi feita em Matlab.

Essa opção deveu-se essencialmente ao facto deste já dispor da implementação do algoritmo genético. Perante isto, foi possível uma mais rápida abordagem ao problema, possibilitando que o foco fosse apenas para o problema da seleção de parâmetros e das características e não a implementação do algoritmo genético.

Apesar da implementação do algoritmo já ser disponibilizada é necessário redefinir a função de cálculo da qualidade dos indivíduos. Seguidamente segue o diagrama de sequência da abordagem usada.

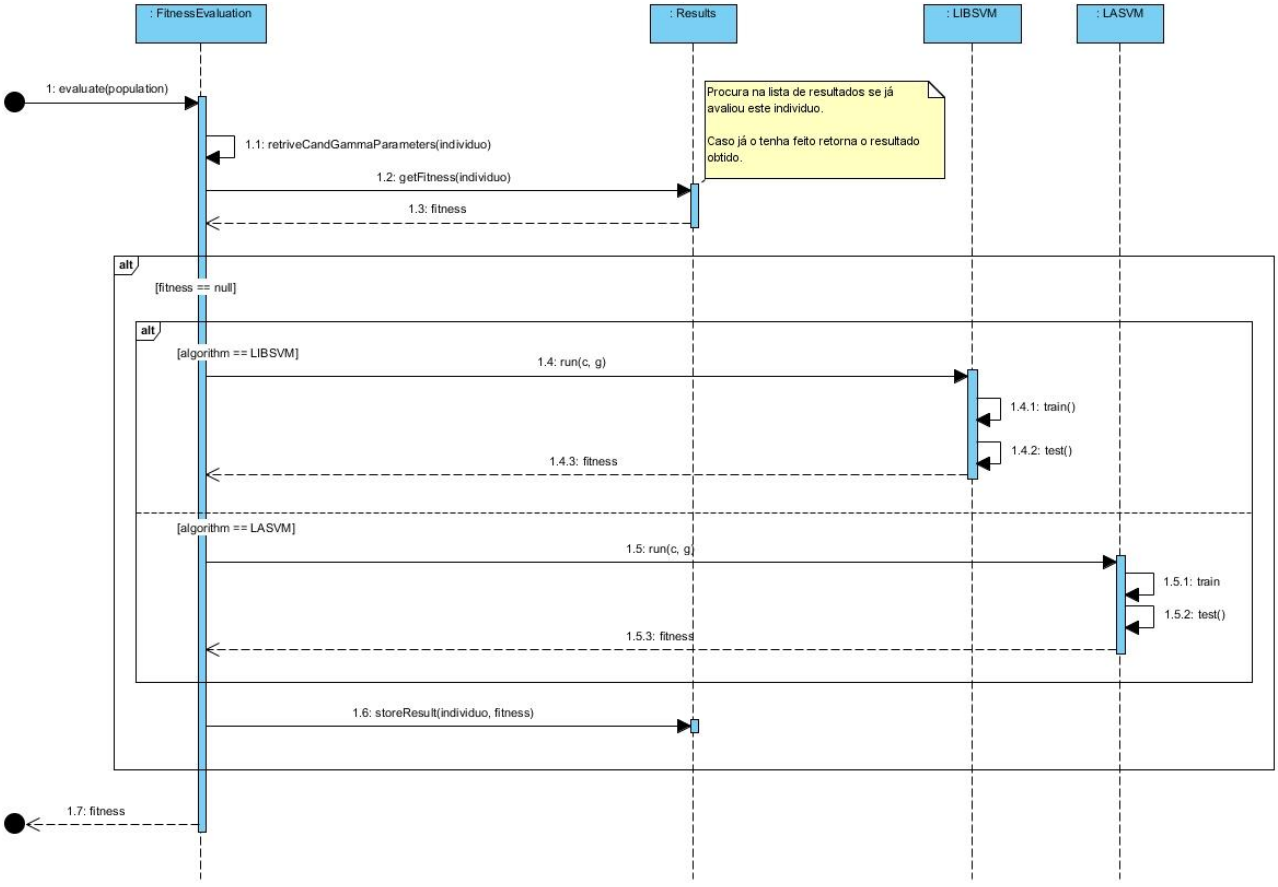


Figura 15 – Diagrama de sequência para avaliação da qualidade dos indivíduos.

Como é espectável que o mesmo individuo seja avaliado várias vezes ao longo da execução do algoritmo genético, optou-se por efetuar o registo do resultado da avaliação de cada individuo. Assim, a cada avaliação verifica-se se esse individuo já foi avaliado e caso já tenha sido devolve-se o resultado obtido. Caso contrário efetua-se o treino e consequente teste para obtermos a qualidade do individuo.

Para guardar os resultados dos vários indivíduos efetua-se a conversão da sequência binária que representa cada um deles e converte-se para decimal, criando um vetor com todas as combinações possíveis.

3.5.2.3. Resultados

Com o intuito de se proceder à validação efetuaram-se testes comparativos entre a abordagem com o algoritmo genético e a pesquisa em grelha. Para isso, utilizou-se um *dataset* de menores dimensões o a1a[5], constituído por 32.561 exemplos, possuindo apenas 123 características.

Na Tabela 1 são apresentados os parâmetros usados para o algoritmo genético, o número de instâncias avaliadas, o número de combinações possíveis e ainda o tempo de execução de cada uma das abordagens.

Tabela 1 – Resultados comparativos e parâmetros usados para a pesquisa em grelha e a otimização através do algoritmo genético.

	GA (1)	GA (2)	GA (3)	Pesquisa em Grelha
Número de indivíduos	10	25	25	
Número de gerações	25	50	50	
Número de gerações executadas	13	26	34	
Taxa de mutação	25%	25%	2%	
Seleção por torneio	2	2	2	
Percentagem de recombinação	80%	80%	80%	
Pontos de corte para recombinação	2	2	2	
Número de bits por parâmetro (C e γ)	4	4	10	
Número de combinações possíveis	256	256	1.048.576	256
Número de instâncias avaliadas	37	121	331	256
Melhor resultado (f-measure):	77,5224%	77,5255%	77,6615%	77,3314%
Tempo de execução	5.42 min	18.15 min	67.55min	42.33 min

[5] Disponível em http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#a1a

Na Figura 16 é possível verificar de uma forma gráfica os resultados em termos do número de instâncias avaliadas, tempo de execução e f-measure. Como se pode constatar a melhor qualidade obtida pelas duas abordagens é muito semelhante, sendo ligeiramente superior com a aplicação do algoritmo genético.

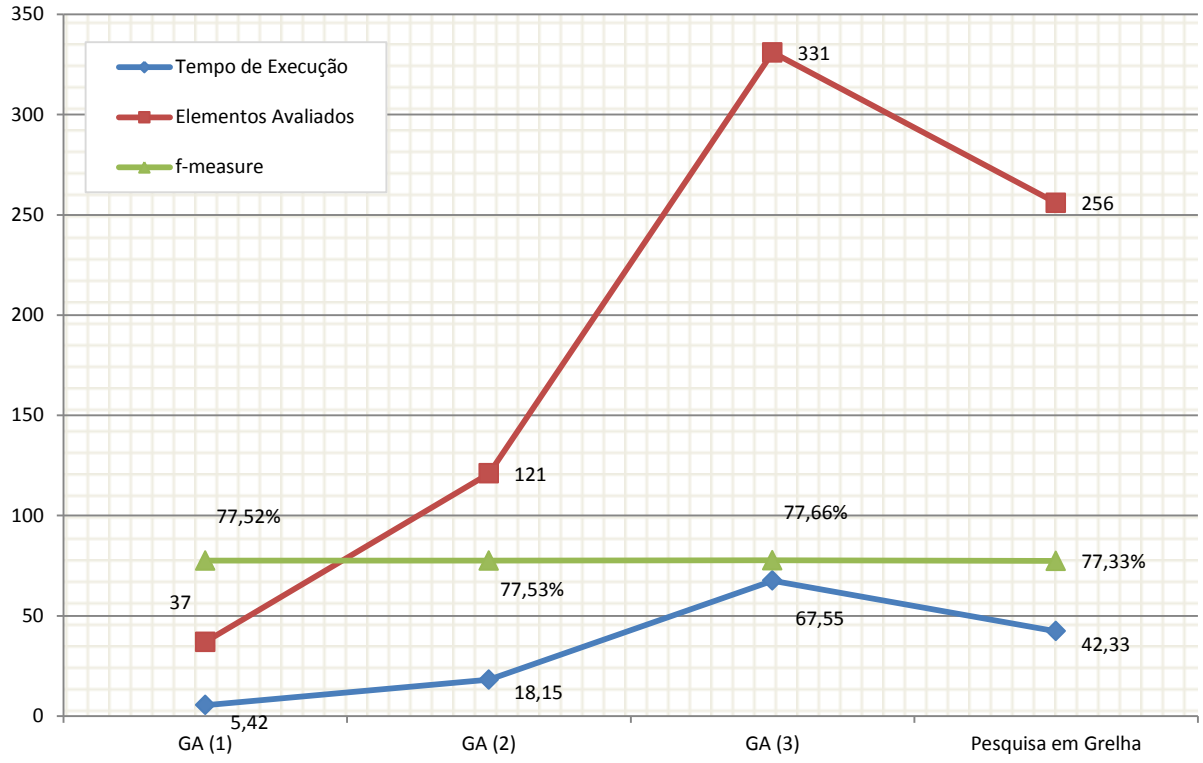


Figura 16 – Comparação de tempos de execução, número de elementos avaliados e a qualidade dos melhores resultados entre a pesquisa em grelha e a otimização através do algoritmo genético.

Após ter-se realizado a comparação com a pesquisa em grelha, optou-se por aplicar o algoritmo para efetuar a otimização de parâmetros e de características simultaneamente. Assim, recorreu-se a 11 cromossomas, 4 para C, 4 para o γ e 3 para identificar o tipo de *n*-gram a ser avaliado. Como algoritmo de aprendizagem supervisionada optou-se pelo LASVM, uma vez que proporciona uma execução mais célere do que o LIBSVM.

Os parâmetros usados são os apresentados na Tabela 2 e na Figura 17 onde é possível visualizar a qualidade dos indivíduos ao longo das várias gerações.

Tabela 2 – Parâmetros usados no algoritmo genético para otimização de parâmetros e características.

Número de indivíduos	10
Número de gerações	20
Número de gerações executadas	13
Taxa de mutação	5%
Seleção por torneio	2
Percentagem de recombinação	80%
Cruzamento em dois pontos	
Número de bits (quatro por parâmetros (C e γ) e três para o tipo de n-gram)	11
Número de combinações possíveis	2048

O melhor resultado foi de 98.127% (f-mesure) com a combinação de três n-grams (1, 2, 3) e com os parâmetros C=32.00 e γ=0.250000).

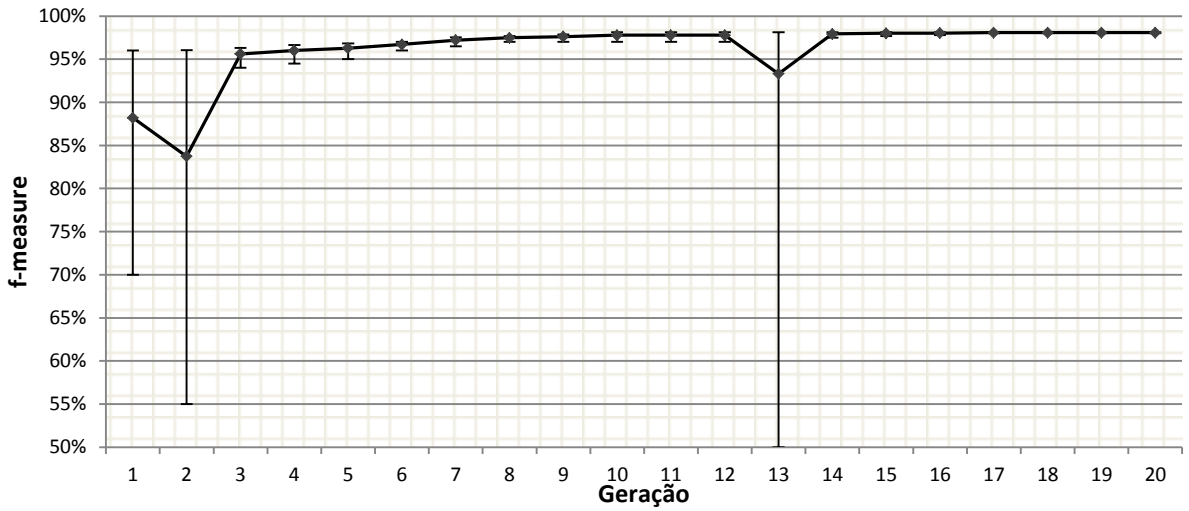


Figura 17 – Resultado da qualidade dos indivíduos (melhor, pior e valor médio).

Apesar de nos testes realizados ter-se obtido uma qualidade ligeiramente superior recorrendo à optimização através do algoritmo genético, optou-se por na restante dissertação fazer uso da pesquisa em grelha. Esta opção deveu-se essencialmente à necessidade de obter mais resultados, sendo para isso necessário efetuar uma avaliação mais ampla e uma maior exploração dos parâmetros do algoritmo. Contudo, o tempo de execução revelou-se o maior entravo à realização de uma avaliação mais ampla.

CAPÍTULO 4. TECNOLOGIAS E FERRAMENTAS

Neste capítulo são apresentadas as ferramentas e linguagens usadas durante a elaboração desta dissertação.

A aplicação desenvolvida para deteção de peptidases, na qual foi utilizada a metodologia apresentada anteriormente, tinha como principal objetivo não ser necessário proceder a qualquer tipo de instalação e ser acessível de qualquer dispositivo com acesso à internet. Assim, optou-se pela elaboração de uma plataforma web. Para tal recorreu-se à tecnologia ASP.NET Web Forms, de forma a construir páginas dinâmicas em HTML e ainda, à utilização de CSS para controlar a aparência da plataforma.

Além destas duas tecnologias, recorreram-se a muitas outras durante a elaboração da dissertação. Seguidamente são apresentadas as que se consideram que tiveram um maior contributo para o resultado final da dissertação.

4.1. WVTool

O WVTool (Wurst 2007) é uma biblioteca desenvolvida em java que permite criar, de uma forma simples e flexível, a representação de um documento num vetor de características. Assim, estes vetores de características posteriormente podem ser usados em vários algoritmos.

Além da criação de vetores de características, esta ferramenta também disponibiliza várias funcionalidades, tais como tokenizers, stemmers e métricas para contabilização das características extraídas.

Nesta dissertação recorreu-se a esta biblioteca essencialmente para proceder à extração de *n*-grams, construção do vetor de características e à contagem dos termos.

4.2. IKVM

Como um dos principais requisitos passava pela disponibilização das metodologias apresentadas numa plataforma web foi escolhida a tecnologia ASP.NET.

No entanto, constatou-se um problema de integração entre as ferramentas desenvolvidas em Java e a tecnologia ASP.NET. Assim, para colmatar a necessidade de incorporar bibliotecas desenvolvidas em java recorreu-se ao IKVM (Frijters).

Através da utilização desta ferramenta é possível compilar código java diretamente para CIL (Common Intermediate Language), facilitando a sua utilização transparente em aplicações .NET.

4.3. JavaScript

O JavaScript é uma linguagem de programação interpretada, essencialmente desenvolvida para permitir uma maior interatividade do que aquela que é possível, recorrendo apenas a HTML. Está essencialmente voltada para o lado do cliente, sendo executada no *browser* do utilizador. Contudo, atualmente este paradigma está a inverter-se, sendo já usado do lado do servidor[6].

Essencialmente no desenvolvimento da plataforma utilizaram-se as seguintes frameworks:

- jQuery[7] (Resig), é uma framework multi-browser que simplifica o desenvolvimento de código javascript;
- jQuery UI[8] (jQuery Foundation), usado para melhorar a interatividade com o utilizador;
- Highcharts[9] (Highsoft), utilizado para criar gráficos.

[6] Node.js server-side software system: http://nodejs.org/
[7] jQuery : http://www.jquery.com/
[8] jQuery Ui : http://jqueryui.com/
[9] Highcharts : http://www.highcharts.com/

4.4. LibSVM

A LibSVM (Chang e Lin 2011) é uma biblioteca desenvolvida com várias finalidades, tais como, classificação, estimativa de distribuição e regressão, disponibilização de formulações de SVMs, classificação de diversas classes e não apenas classificação binária, entre outras. A escolha da utilização desta biblioteca partiu do facto de ser amplamente utilizada, permitindo assim efetuar comparações de resultados com outros estudos e com outros algoritmos.

Esta biblioteca está disponível em diversas linguagens de programação que vão desde o C, Python, R, Ruby, Haskel, Cuda entre outras. Neste trabalho optou-se pela utilização da biblioteca implementada em C++ versão 3.1[10].

4.5. LASVM

O algoritmo incremental LASVM é um classificador que usa aproximação online e que obtém resultados semelhantes à implementação de uma SVM, efetuando apenas uma única passagem sequencial através dos exemplos de treino. Além disso, apresentou ganhos significativos ao nível da memória e do tempo de treino.

Devido a estes fatores optou-se pela sua utilização e avaliação ao longo deste trabalho, tendo sido usada a versão 1.1[11]. Esta versão dispõe de uma biblioteca implementada em C com a implantação das funcionalidades base, *kernel* e métodos de processamento e reprocessamento. Para além disso ainda disponibiliza dois programas desenvolvidos em C++ que permitem efetuar treino e teste de modelos.

[10] LibSVM : http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[11] LASVM : http://leon.bottou.org/projects/lasvm

4.6. Matlab

MATLAB é uma linguagem de alto nível com um ambiente interativo para computação numérica, visualização e programação. Possui uma vasta quantidade de *toolboxes* específicos para várias áreas, nomeadamente, otimização, processamento de imagens e de sinal, entre muitas outras.

Nesta dissertação o Matlab foi utilizado essencialmente para seleção de parâmetros através do algoritmo genético, pois este já dispõe de uma implementação do algoritmo.

A utilização do Matlab para esse fim deveu-se essencialmente a uma mais rápida prototipagem e validação da metodologia e ainda à confiança dada pela implementação do algoritmo.

4.7. MEROPS

O *MEROPS* é uma base de dados desenvolvida por Rawlings, Barrett et al. (2012) com o intuito de conter toda a informação relativa à classificação e nomenclatura das peptidases e dos seus inibidores, permitindo a sua consulta e constante atualização.

Nesta base de dados as peptidases foram catalogadas manualmente, sendo os substratos e inibidores organizados hierarquicamente em espécies, famílias e clãs. Um clã incluiu todas as peptidases que divergiram da mesma origem evolucionária. Por outro lado, as famílias contem as peptidases que possuem uma sequência de aminoácidos similares. Por último, a espécie refere-se à peptidase propriamente dita.

A distinção entre as peptidases é realizada através da sua especificidade, de acordo com a sequência que cliva.

O MEROPS além de conter as proteínas catalogadas possui também representações em três dimensões das estruturas das peptidases.

Esta base de dados foi usada nesta dissertação para a recolha das peptidases necessárias para a construção do conjunto de dados, que foi posteriormente utilizado na avaliação da metodologia para deteção das mesmas.

4.8. SCOP

SCOP (Murzin, Brenner et al. 1995) é uma base de dados de classificação de proteínas que tem por objetivo determinar relações evolucionárias entre as proteínas. A sua criação ocorreu em 1994 e foi elaborada a partir das características dos domínios estruturais das proteínas, nomeadamente sequência de aminoácidos e estrutura.

Nesta base de dados as proteínas são agrupadas consoante as classes, grupos, superfamílias, famílias, domínios proteicos e espécies. Os grupos estão agrupados em classes e estas mesmas são a raiz da hierarquia de classificação do SCOP. Esta classificação é realizada manualmente.

Esta base de dados de proteínas foi usada nesta dissertação essencialmente para a recolha de exemplos de proteínas não peptidases.

CAPÍTULO 5. CASO DE ESTUDO: DETEÇÃO DE PEPTIDASES

Neste capítulo pretende-se apresentar a utilidade e aplicação da metodologia referida anteriormente ao problema de deteção de peptidases.

Assim, inicia-se com a apreciação da metodologia apresentada, avaliando o tipo de características e as métricas de contabilização das mesmas. Após a avaliação preliminar, a metodologia é então aplicada a um *dataset* de maiores dimensões e estima-se os resultados obtidos através da aplicação de três algoritmos distintos de classificação supervisionados C-SVC e One-Class, e LASVM incremental.

5.1. Avaliação Preliminar

Com o intuito de se proceder à validação da capacidade discriminativa das características baseadas em *text mining* e das métricas de contabilização das mesmas, verificou-se quais as que possuem uma capacidade de distinção mais promissora.

Efetuou-se a validação das várias características e métricas de contabilização através de um *dataset* para deteção de peptidases de menor dimensão introduzido por Morgado, Pereira et al. (2010). Tal foi executado de forma a obter-se uma avaliação geral da aplicabilidade desta metodologia o mais rápido possível, permitindo escolher e melhorar a metodologia para posteriormente aplicar a *datasets* mais alargados.

Assim o *dataset* usado dispõe de 6.006 proteínas, 3.003 peptidases do *MEROPS* (Rawlings, Barrett et al. 2012) 8.5 e 3.003 não-peptidases do SCOP (Murzin, Brenner et al. 1995) 1.75, recolhidas de forma aleatória e seguidamente divididas em treino e teste como apresentado na Tabela 3.

Tabela 3 – Divisão do *dataset* reduzido para deteção de peptidases.

	Núm. de proteínas	
	Train	**Test**
Peptidases	2002	1001
Não Peptidases	2002	1001

A partir deste *dataset* foram extraídas as principais características através de técnicas de *text mining* referidas anteriormente. Foram feitas extrações de quatro tipos de *n*-grams, onde *n* variou de 1 a 4 (ver detalhes na secção 3.1 – Extração de Características).

Para cada uma dessas extrações efetuou-se a contabilização das características com os quatro tipos de métricas apresentadas anteriormente, tendo-se avaliado 16 combinações (4 *n*-gram e 4 métricas para calcular a relevância das características). Na Tabela 4 é apresentado o número de características extraídas por *n*-gram. Como é natural o número de características varia apenas por *n*-gram.

Tabela 4 – Combinações usadas para a avaliação preliminar e número de características extraídas.

	Núm. características
n-gram (1)	20
n-gram (2)	400
n-gram (3)	4645
n-gram (4)	157

Com o aumento do tamanho dos *n*-grams verificou-se que existe um crescimento exponencial no número de características extraídas. Desse modo, optou-se por aplicar *prunning* para limitar o número de características.

5.1.1. Resultados

Para demonstrar a validade da abordagem evidenciada anteriormente procedeu-se ao treino e validação dos resultados com o classificador C-SVC, disponibilizado pela biblioteca LibSVM. Para obterem-se resultados mais fidedignos optou-se pela aplicação de uma pesquisa em grelha para os parâmetros *C* (2^{-4} *a* 2^{11}) e γ(2^{-13} *a* 2^{2}) do *kernel* RBF.

Esta pesquisa em grelha efetuará a avaliação de 256 combinações por cada uma das 16 combinações apresentadas na secção anterior, perfazendo um conjunto de 4.096 treinos e testes. Na Figura 18 são apresentados os melhores resultados obtidos para as 16 combinações.

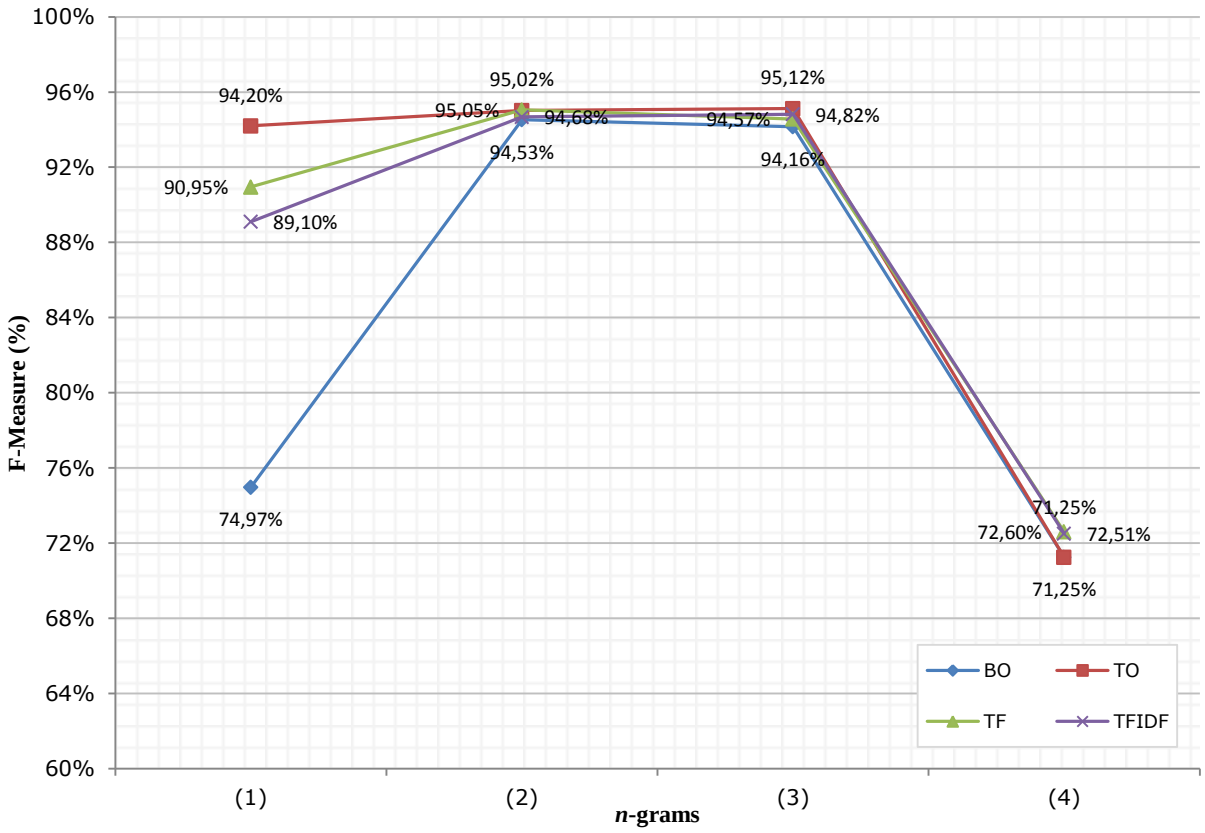


Figura 18 – Melhores resultados com o *dataset* preliminar para as 16 combinações.

Como é possível verificar na figura acima, em termos de métricas de contabilização da relevância de cada uma das características, não existe uma grande discrepância entre estas, existindo apenas uma menor capacidade discriminativa com a utilização de métrica que indica apenas se a característica está presente (BO).

Relativamente aos tipos de *n*-grams verificou-se que com o aumento do tamanho dos n-gram a capacidade discriminativa diminui, constatável principalmente nos resultados apresentados com a utilização de *n*-grams de 4. Isto deve-se essencialmente à forma rígida de extração das características, onde todos os aminoácidos de um *n*-gram têm de ser iguais para poderem ser contabilizados. Também foi possível verificar esse facto através do número de características, já que este cresce exponencialmente com o aumento do tamanho do *n*-gram.

Devido a esse crescimento exponencial, foi necessário aplicar *pruning* à extração de *n*-grams de 4 para tentar obter as características mais discriminativas. Daí o número de características ter reduzido substancialmente. Contudo, note-se que possui mais características que o *n*-gram de 1, mas mesmo assim obteve um resultado bastante inferior, denotando que as características obtidas têm uma baixa capacidade discriminativa.

Fazendo uma avaliação geral da aplicabilidade desta metodologia, verificou-se que esta permite obter resultados promissores, já que os resultados obtidos aproximam-se dos apresentados por Morgado, Pereira et al. (2010) (ver Tabela 5) para deteção de peptidases.

Tabela 5 – Comparação com os resultados reportados por Morgado, Pereira et al. (2010) para deteção de peptidases.

Algoritmo	# características	Accuracy	Sensitivity	Specificity	Precision
SVM *(kernel RBF)* (Morgado, Pereira et al. 2010)	2059	95,65%	96,00%	95,00%	96,00%
SVM-RFE *(kernel RBF)* (Morgado, Pereira et al. 2010)	148	95,65%	96,00%	95,00%	96,00%
PSI-BLAST (Morgado, Pereira et al. 2010)		93,25%	100,00%	87,00%	88,00%
***n*-gram (2) - TF**	400	95,05%	94,91%	95,20%	95,19%

Relativamente aos parâmetros, dependendo do tipo de métrica de contabilização, estes têm variações distintas, como pode-se verificar na Figura 19 e Figura 20.

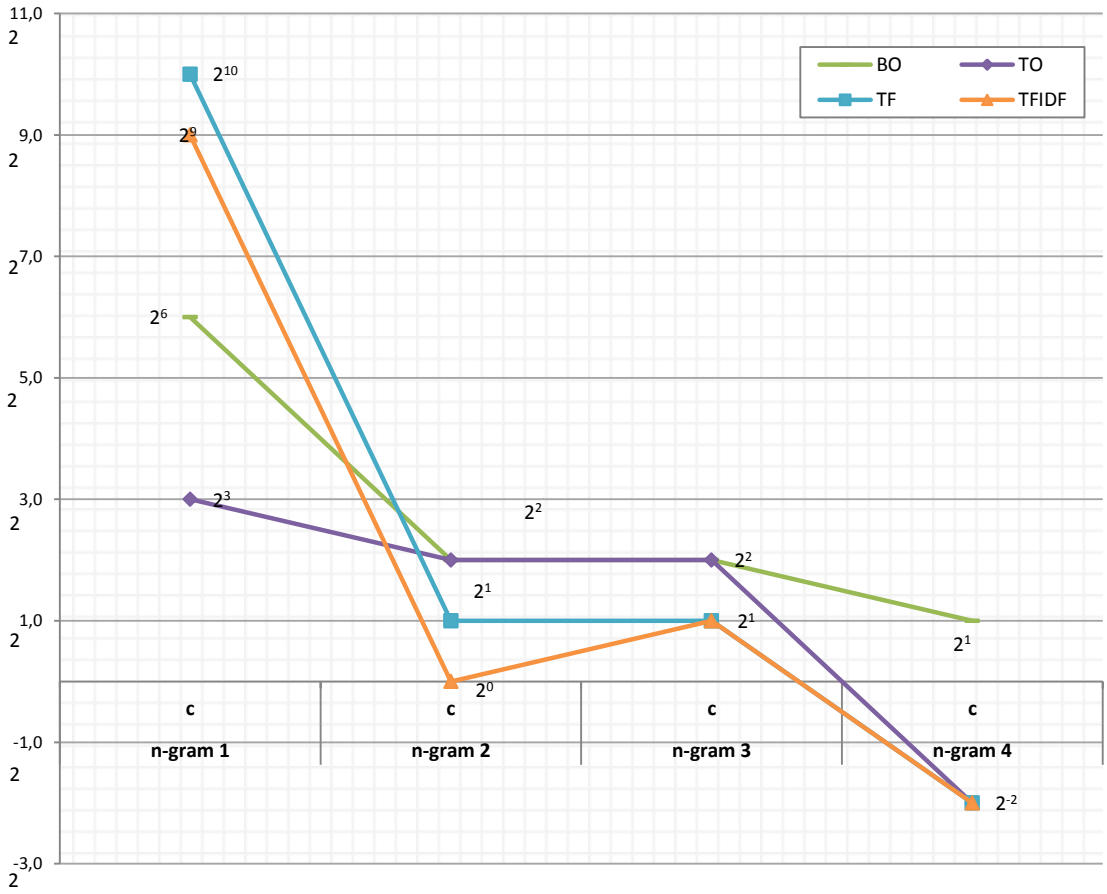


Figura 19 - Variação dos valores dos parâmetros de C para as várias combinações.

Como era expectável, os parâmetros de TF e TFIDF são bastante semelhantes, isto deve-se ao facto do valor de a variação destas duas métricas ser igual (entre 0 e 1). A métrica que demonstra uma maior discrepância nos valores é a TO, uma vez que esta contabiliza as ocorrências de uma determinada característica, fazendo com que existam discrepâncias bastante marcadas entre os valores das características.

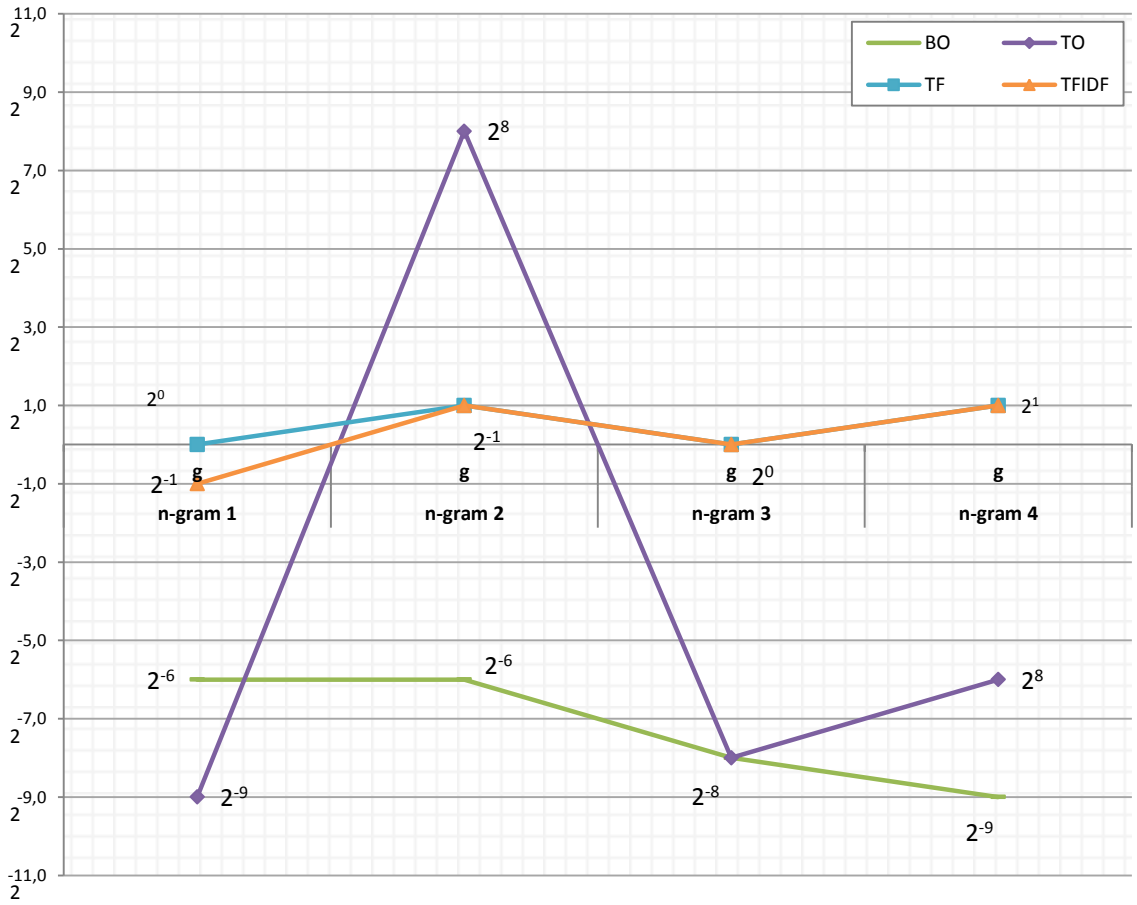


Figura 20 - Variação dos valores dos parâmetros de **γ** para as várias combinações.

Uma apresentação mais detalhada dos resultados obtidos e parâmetros usados são apresentados em formato tabular em anexo (ver Anexo C).

Em avaliações futuras optou-se por efetuar extrações apenas de *n*-grams de 1,2 e 3, sendo utilizado o TF como métrica de contabilização das características.

5.2. Avaliação Experimental

Após ter-se validado a metodologia na avaliação preliminar e ter-se verificado quais as métricas e os tipos de características que se deveria dar mais atenção, procedeu-se à ampliação do estudo com o intuito de melhorar e estender os resultados apresentados. Assim, efetuou-se a análise de mais classificadores e alargou-se o *dataset* utilizado.

5.2.1. Conjunto de Dados

De forma a elaborar uma avaliação mais exaustiva da deteção de peptidases, procedeu-se à construção de um *dataset* mais alargado, sendo este constituído por 20.778 sequências de proteínas (detalhado na Tabela 6 – Detalhes do dataset para deteção de peptidases.).

A partir do *MEROPS* 9.4 (Rawlings, Barrett et al. 2012) recolheu-se 10% das sequências classificadas como peptidases (18.068) e 2.710 não-peptidases foram extraídas do SCOP (Murzin, Brenner et al. 1995), tendo ambas as amostras sido recolhidas de forma aleatória.

Na elaboração do *dataset* foi também tido em conta as diferentes quantidades de sequências existentes por família. Assim, foram recolhidas sequências de todas as famílias de forma proporcional.

Tabela 6 – Detalhes do *dataset* para deteção de peptidases.

	MEROPS	Dataset
Peptidase		
Aspartic (A)	8.478	848
Cysteine (C)	29.105	2.911
Glutamic (G)	84	9
Metallo (M)	63.853	6.385
Asparagine (N)	409	41
Serine (S)	69.362	6.936
Threonine (T)	5.906	590
Unknown (U)	3.480	348
Não-Peptidase		2.710
Total		20.778

Aquando da elaboração do *dataset*, verificou-se a existência de um problema que se devia essencialmente à vasta quantidade de proteínas classificadas como peptidases (exemplos positivos) e uma pequena quantidade de proteínas classificadas como não sendo peptidases.

Assim, seria necessário alargar o número de exemplos negativos, de forma a equilibrar o *dataset*. Contudo, fez-se uso da dificuldade relativa à seleção de exemplos negativos para se optar por uma outra abordagem para este problema que passaria pela utilização de um classificador One-Class, permitindo assim recorrer apenas a exemplos positivos.

Desse modo, o *dataset* apresentado anteriormente foi dividido em treino e teste, sendo que o treino apenas dispõe de exemplos positivos.

A divisão do dataset é detalhada na Tabela 7, tendo-se optado por dividir os dados de forma a obter a mesma quantidade de exemplos positivos e negativos no conjunto de teste.

Tabela 7 – Detalhes do dataset para deteção de peptidases para algoritmo One-Class.

	Treino	Teste
Peptidase		
Aspartic (A)	721	127
Cysteine (C)	2474	437
Glutamic (G)	7	2
Metallo (M)	5427	958
Asparagine (N)	35	6
Serine (S)	5896	1040
Threonine (T)	502	88
Unknown (U)	296	52
Não Peptidase		
		2.710
Total	15.358	5.420

Na secção seguinte são apresentados os resultados comparativos entre o algoritmo C-SVC RBF e o One-Class.

5.2.2. Resultados

Como já mencionado anteriormente, antes de se proceder à aplicação de um algoritmo de classificação é necessário extrair as principais características dos dados que se pretendem classificar. Para o efeito, efetuou-se a extração das características do *dataset* utilizando-se técnicas de *text mining* previamente descritas.

Foram criados sete tipos de extrações através da criação de todas as combinações possíveis para os três tipos de *n*-grams: (*i*) unigram, (*ii*) bigram e (*iii*) trigram. Sendo utilizada a frequência dos termos (TF) como métrica de contabilização da relevância das características. Os detalhes dessas extrações e as combinações criadas são apresentados na Tabela 8.

Tabela 8 – Detalhes das características extraídas do *dataset* para deteção de peptidases.

Tipo de Extração	# características
n-gram 1 (unigram)	24
n-gram 2 (bigram)	473
n-gram 3 (trigram)	3378
n-gram 1 e 2	497
n-gram 1 e 3	3402
n-gram 2 e 3	3851
n-gram 1, 2 e 3	3875

Após a obtenção das extrações efetuou-se a avaliação dos resultados para o C-SVC com o *kernel* gaussiano (RBF) através do *cross-validation* de 5 *folds* e recorrendo ao *dataset* completo apresentado na Tabela 6 – Detalhes do dataset para deteção de peptidases. Para o algoritmo One-Class, utilizou-se o *dataset* dividido, apresentado na Tabela 7, já que este aceita apenas exemplos positivos no processo de treino de um modelo.

A escolha dos parâmetros foi novamente efetuada através de uma pesquisa em grelha e os detalhes dessa pesquisa são apresentados na Tabela 9.

Tabela 9 – Detalhes da pesquisa em grelha para deteção de peptidases C-SVC e One-Class.

Parâmetro			
	C	$2^{-4}, 2^{-3} \ldots 2^{11}$	16
	γ	$2^{-13}, 2^{-12} \ldots 2^{2}$	16
	nu	0.1, 0.2, …0.9	9

O *kernel* RBF do algoritmo C-SVC necessita que dois parâmetros sejam ajustados C e γ, como foi efetuado na secção 5.1 – Avaliação Preliminar. Assim, foram executados 256 treinos e testes por extração, e posteriormente foram escolhidos os parâmetros que apresentavam melhor qualidade.

Já o algoritmo One-Class necessita que sejam variados os parâmetros γ e *nu*, tendo sido necessário apenas recorrer à execução de 144 treinos e testes por extração. Na Figura 21 podem ser visualizados os melhores resultados obtidos após a pesquisa em grelha para os dois algoritmos.

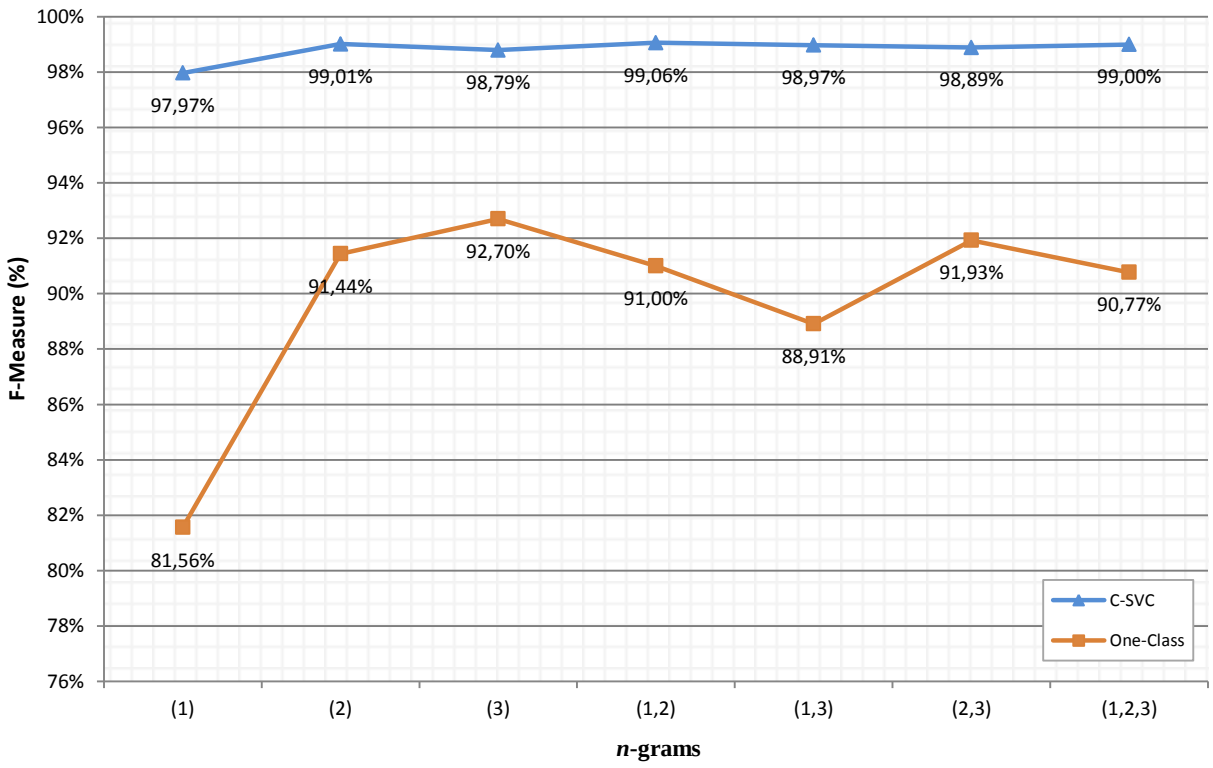


Figura 21 - Melhores resultados F-Measure para os algoritmos C-SVC (Kernel RFB) e One-class

Os resultados obtidos demonstram que o algoritmo C-SVC com o kernel RBF consegue melhores resultados, sendo que a combinação dos *n*-grams 1 e 2 consegue superar em 0,048% os resultados obtidos com *n*-grams de 2.
Os parâmetros utilizados e as restantes métricas de avaliação podem ser consultadas na Tabela 10 e Tabela 11.

Tabela 10 - Detalhes dos melhores resultados obtidos com o algoritmo C-SVC kernel RBF.

	c	γ	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**Recall**	**F-Measure**
n-gram (1)	2^3	2^2	96,434%	98,793%	80,701%	97,153%	98,793%	97,966%
n-gram (2)	2^1	2^1	98,277%	99,181%	92,251%	98,842%	99,181%	99,011%
n-gram (3)	2^0	2^0	97,892%	99,303%	88,487%	98,291%	99,303%	98,794%
n-gram (1-2)	2^7	2^{-3}	98,360%	99,290%	92,140%	98,830%	99,290%	99,059%
n-gram (1-3)	2^0	2^0	98,200%	99,491%	89,594%	98,456%	99,491%	98,971%
n-gram (2-3)	2^0	2^{-1}	98,056%	99,181%	90,554%	98,592%	99,181%	98,886%
n-gram (1-2-3)	2^0	2^0	98,248%	99,319%	91,107%	98,675%	99,319%	98,996%

O algoritmo One-Class demonstrou ficar abaixo do C-SVC entre 6% e 16%, contudo há que ressalvar o facto de que este algoritmo apenas usou exemplos positivos. Assim, apesar dos resultados serem substancialmente inferiores aos obtidos pelo algoritmo de C-SVC, esta metodologia deve ser tida em consideração para futuras utilizações em *datasets*, para o qual não seja possível ou desejável identificar exemplos negativos.

Em relação ao tipo de característica que apresentou melhores resultados verificou-se que neste caso são os *n*-grams de 3.

Todos os detalhes sobre os parâmetros usados e as restantes métricas são apresentados na Tabela 11.

Tabela 11 - Detalhes dos melhores resultados obtidos com o algoritmo One-Class.

	nu	γ	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**Recall**	**F-Measure**
n-gram (1)	0,2	2^2	81,94%	79,89%	83,99%	83,30%	79,89%	81,56%
n-gram (2)	0,1	2^0	91,59%	89,85%	93,32%	93,08%	89,85%	91,44%
n-gram (3)	0,1	2^{-11}	92,62%	93,73%	91,51%	91,70%	93,73%	92,70%
n-gram (1-2)	0,1	2^1	91,18%	89,15%	93,21%	92,92%	89,15%	91,00%
n-gram (1-3)	0,1	2^1	89,23%	86,35%	92,10%	91,62%	86,35%	88,91%
n-gram (2-3)	0,1	2^{-13}	91,88%	92,44%	91,33%	91,42%	92,44%	91,93%
n-gram (1-2-3)	0,50	2^0	91,00%	88,56%	93,43%	93,10%	88,56%	90,77%

5.3. Avaliação Algoritmo Incremental (LASVM)

Nesta secção pretende-se apresentar a avaliação feita com o algoritmo LASVM, expor-se o *dataset* usado e os resultados obtidos com o mesmo.

5.3.1. Dados de treino

De forma a avaliar o algoritmo LASVM procedeu-se à aplicação do *dataset* utilizado anteriormente (apresentado na Tabela 6). Contudo, como este algoritmo não tem implementado o *cross-validation*, foi necessário efetuar a divisão do *dataset* em treino e teste (ver Tabela 12).

Tabela 12 - *Dataset* de sequências dividido para treino e teste.

	Treino	**Teste**
Peptidase		
Aspartic (A)	721	127
Cysteine (C)	2.474	437
Glutamic (G)	7	2
Metallo (M)	5.427	958
Asparagine (N)	35	6
Serine (S)	5.896	1.040
Threonine (T)	502	88
Unknown (U)	296	52
Não Peptidase	1.806	904
Total	17.164	3.614

5.3.2. Resultados

Fazendo uso do *dataset* acima apresentado procedeu-se ao treino e teste com o algoritmo LASVM e posteriormente, compararam-se os resultados com os obtidos com o algoritmo da biblioteca LIBSVM (C-SVC com *kernel* RBF) (ver Figura 22).

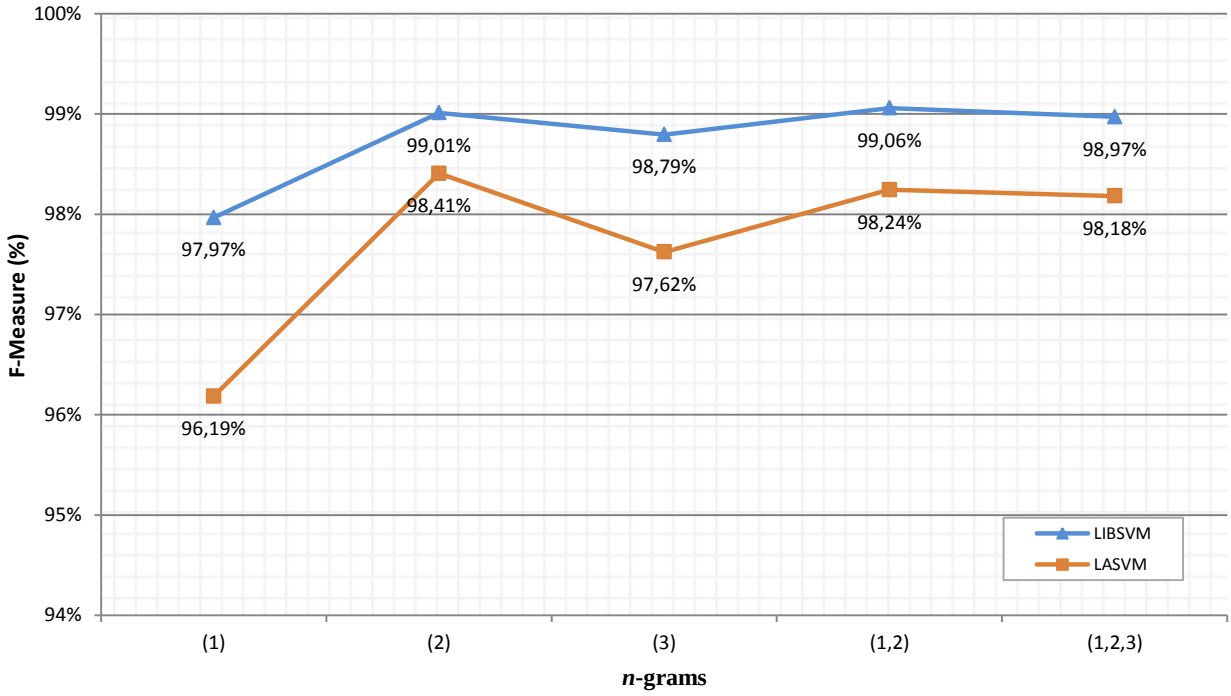


Figura 22 – Comparação de resultados entre o algoritmo LASVM e C-SVC.

Como se pode constatar o algoritmo LASVM não melhora os resultados conseguidos com o algoritmo C-SVC. Apesar disto, os resultados apresentados são bastante próximos. De salientar que o tipo de *n*-gram que apresenta melhores resultados passou a ser o *n*-gram de 2.

Já que o algoritmo LASVM possui um método designado por FINISH Step que permite que no final da construção do modelo seja feita uma redução do número de vetores de suporte (SV), procedeu-se à comparação da complexidade dos modelos gerados.

Como se pode verificar na Figura 23, o LASVM apresenta uma menor complexidade nos modelos gerados (um menor número de vetores de suporte), tornando-se ainda mais evidente quando é utilizado o FINISH Step.

Além disso, verificou-se também que o algoritmo LASVM apresenta uma maior distinção, quando a complexidade dos *datasets* aumenta. Isto é visível, porque com *n*-gram de 1 a LIBSVM (C-SVC com *kernel* RBF) apresenta uma complexidade comparável, contudo quando o número de características do *dataset* aumenta (ex. *n*-gram de 3) o LASVM apresenta uma diminuição da complexidade bastante assinalável.

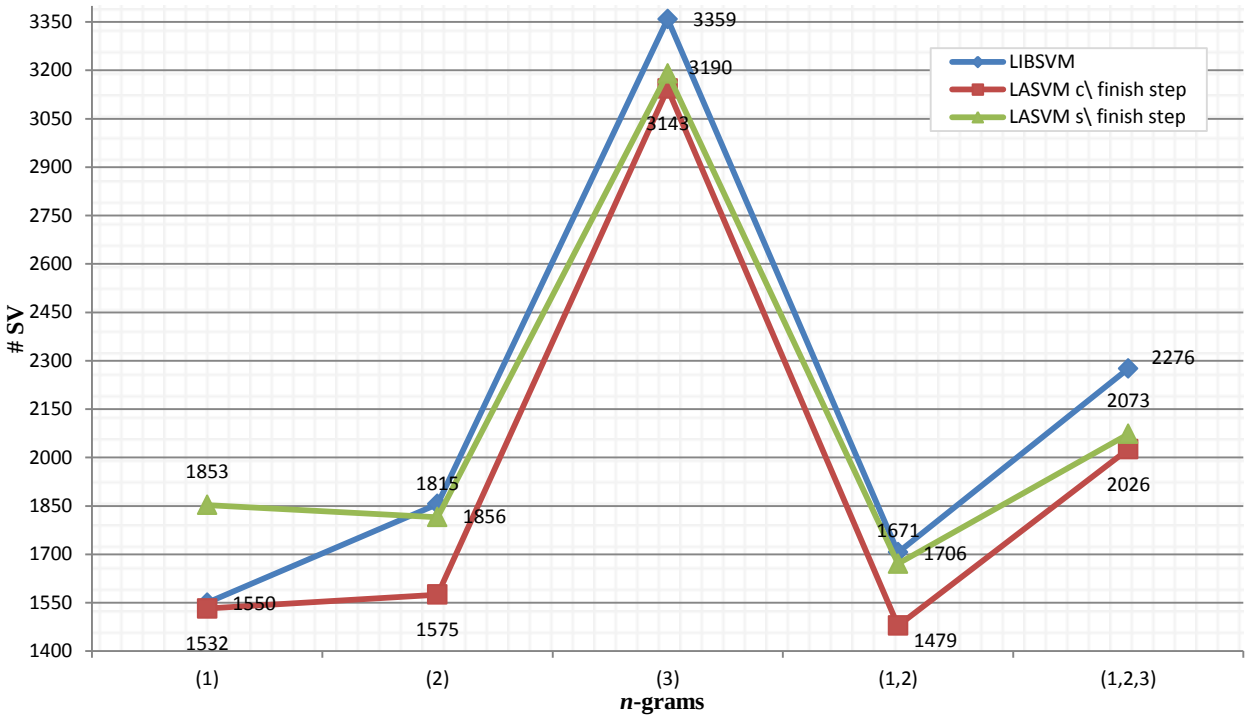


Figura 23 – Comparação da complexidade dos modelos gerados entre LASVM e C-SVC (kernel RBF).

Seguidamente, procedeu-se à comparação dos tempos de treino destes dois algoritmos. Para isso, definiu-se os mesmos parâmetros de C e γ (ver Tabela 13) para ambos os algoritmos, para que estes não tivessem influência no tempo de treino.

Tabela 13 – Parâmetros usados para avaliar o tempo de treino dos algoritmos LASVM e C-SVC.

	c	γ
n-gram (1)	1000	0,562341325
n-gram (2)	100	0,177827941
n-gram (3)	10	0,562341325
n-gram (1,2)	100	0,177827941
n-gram (1,2,3)	100	0,177827941

Constatou-se que o tempo de treino apresenta o mesmo comportamento que a complexidade dos modelos apresentada anteriormente. Uma vez que quando a complexidade do *dataset* em estudo é menor, a LIBSVM consegue terminar a execução do algoritmo mais rapidamente, contudo quando a complexidade do *dataset* aumenta o LASVM começa a destacar-se e apresenta melhorias bastante significativas. Por exemplo, *n*-gram de 3 consegue construir um modelo em menos 29 min do que a LIBSVM com uma diferença em termos de F-Measure de 1,172%.

Em relação à diferença de tempos fazendo uso do FINISH Step, constatou-se que o tempo de treino acrescido compensará aquando da utilização do modelo.

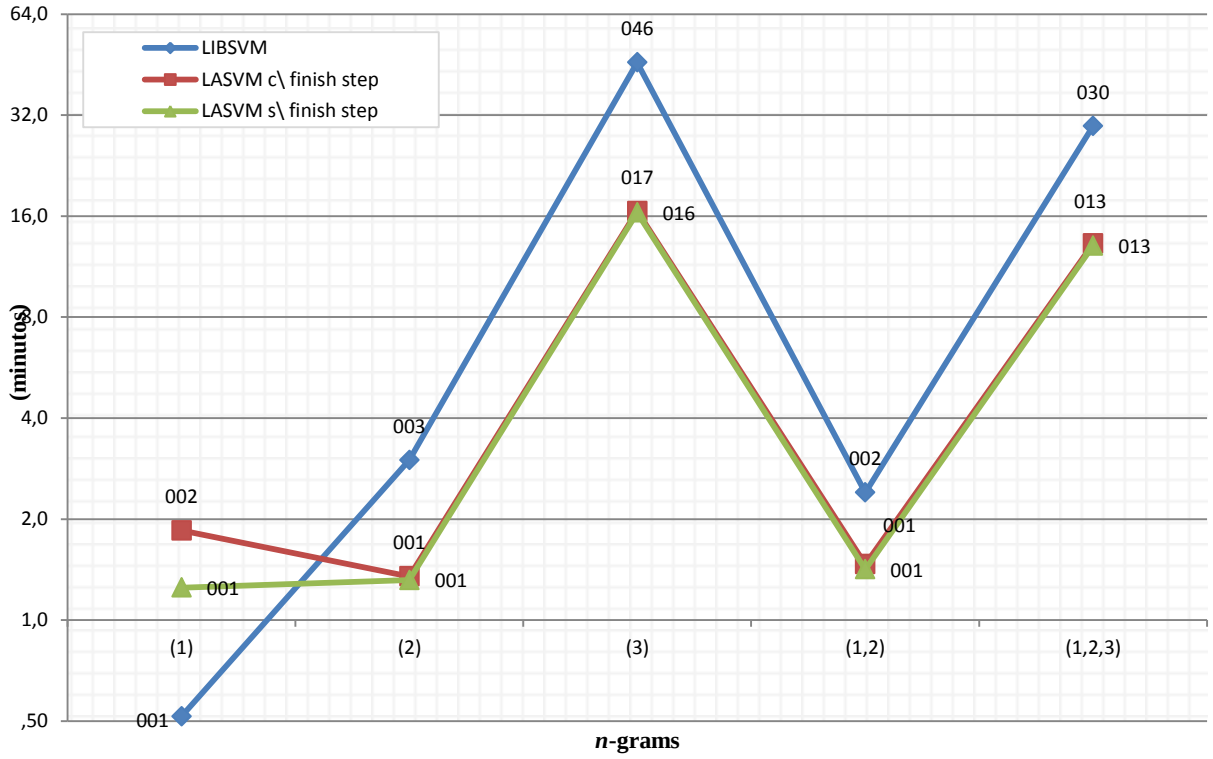


Figura 24 - Comparação dos tempos de treino entre LASVM e C-SVC (kernel RBF).

5.4. Validação da Metodologia

Nesta secção pretende-se validar a utilização da metodologia apresentada para um novo caso de estudo, a identificação de interações entre proteínas. Neste sentido, pretende-se comprovar que esta metodologia é válida e aplicável a outros problemas.

Na abordagem a este caso de estudo foi necessário proceder a uma pequena alteração à metodologia apresentada anteriormente, já que neste caso existem duas proteínas por exemplo.

Assim, decidiu-se proceder à extração das características separadamente, isto é, extraíram-se as características da primeira e da segunda sequência de aminoácidos, seguidamente agruparam-se as características e no final procedeu-se ao cálculo da relevância como se fossem apenas uma sequência.

A utilização desta abordagem deve-se ao facto de não serem gerados *n*-grams com o final da primeira proteína e com o início da segunda. Na Figura 25 é ilustrado esse problema, sendo que em *a)* é possível ver que são gerados apenas sete bigrams distintos, enquanto que em *b)* são gerados oito. A subsequência de aminoácidos sublinhada é gerada inadvertidamente sendo composta pelo último aminoácido da primeira proteína e o primeiro da segunda.

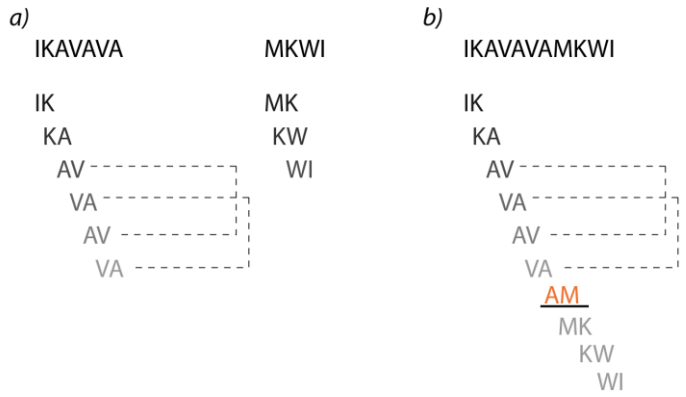


Figura 25 - Extração de *n*-grams de duas sequências de aminoácidos. *a*) extração separada, *b*) extração após junção das duas sequências.

5.4.1. Conjunto de Dados

Para este caso de estudo foi utilizado um *dataset* introduzido por Chen e Liu (2005), sendo posteriormente utilizado por Zaki, Lazarova-Molnar et al. (2009). Este é constituído por 15.409 pares de proteínas do microrganismo *Yeast* com interações entre si, que foram recolhidas da base de dados DIP (Database of Interacting Proteins) (Salwinski, Miller et al. 2004). Sendo que 5.719 desses pares foram recolhidos por Deng, Mehta et al. (2002) e 2.238 por Schwikowski, Uetz et al. (2000). Posteriormente, esses conjuntos de pares de proteínas foram combinados, tendo sido removidos os pares iguais e obtido uma amostra de 9.834 pares de proteínas com interações.

Por não existir um conjunto de proteínas que não interagem entre si, existiu a necessidade de gerados 8.000 exemplos de forma aleatória. Tendo por base a premissa que os pares de proteínas que não estão no conjunto de proteínas que interagem, são proteínas que não interagem.

Este *dataset* já se encontrava dividido em teste e treino (ver Tabela 14), sendo que cada um deles com 8.917, onde 4.917 são exemplos positivos e 4000 são exemplos negativos.

Tabela 14 – *Dataset* Interação entre Proteínas (Zaki, Lazarova-Molnar et al. 2009).

	Treino	**Teste**
Pares de proteínas com interação (exemplos positivos)	4.917	4.917
Pares de proteínas sem interação (exemplos negativos)	4.000	4.000
Total	8.917	8.917

Zaki et al. disponibiliza o *dataset*[12] em três ficheiros de texto: um com a lista de sequências de proteínas em formato FASTA e dois com os pares de identificadores das proteínas, um para treino e outro para teste.

[12] http://faculty.uaeu.ac.ae/nzaki/PPI_PS.htm

5.4.2. Resultados Preliminares

Utilizando o *dataset* apresentado acima procedeu-se à extração das características, aplicando as técnicas de *text mining* previamente descritas.

Dessa forma, foram criados três tipos de extrações com os *n*-grams: (*i*) unigram, (*ii*) bigram e (*iii*) trigram, usando a frequência dos termos (TF) como métrica de contabilização da relevância das características. Os detalhes dessas extrações e as combinações criadas são apresentados na Tabela 15.

Tabela 15 – Detalhes das características extraídas do *dataset* de PPI.

Tipo de Extração	# características
n-gram 1 (unigram)	20
n-gram 2 (bigram)	51
n-gram 3 (trigram)	5.310

Utilizando as três extrações apresentadas acima, procedeu-se ao treino e teste com o algoritmo C-SVC com o *kernel* RBF através da aplicação de uma pesquisa em grelha com os parâmetros C (2^{-4}, 2^{-3} … 2^{11}) e γ (2^{-13}, 2^{-12} … 2^{2}).

Como é possível verificar na Figura 26, com a abordagem apresentada conseguiu-se obter resultados similares aos apresentados por Zaki, Lazarova-Molnar et al. (2009). Assim, demostra-se que a metodologia usada apesar de simples consegue obter resultados promissores.

Contudo, é de salientar que estes resultados são preliminares, pretendendo-se validar a metodologia noutro caso de estudo. Assim, este caso de estudo deverá ser alvo de um estudo mais alargado.

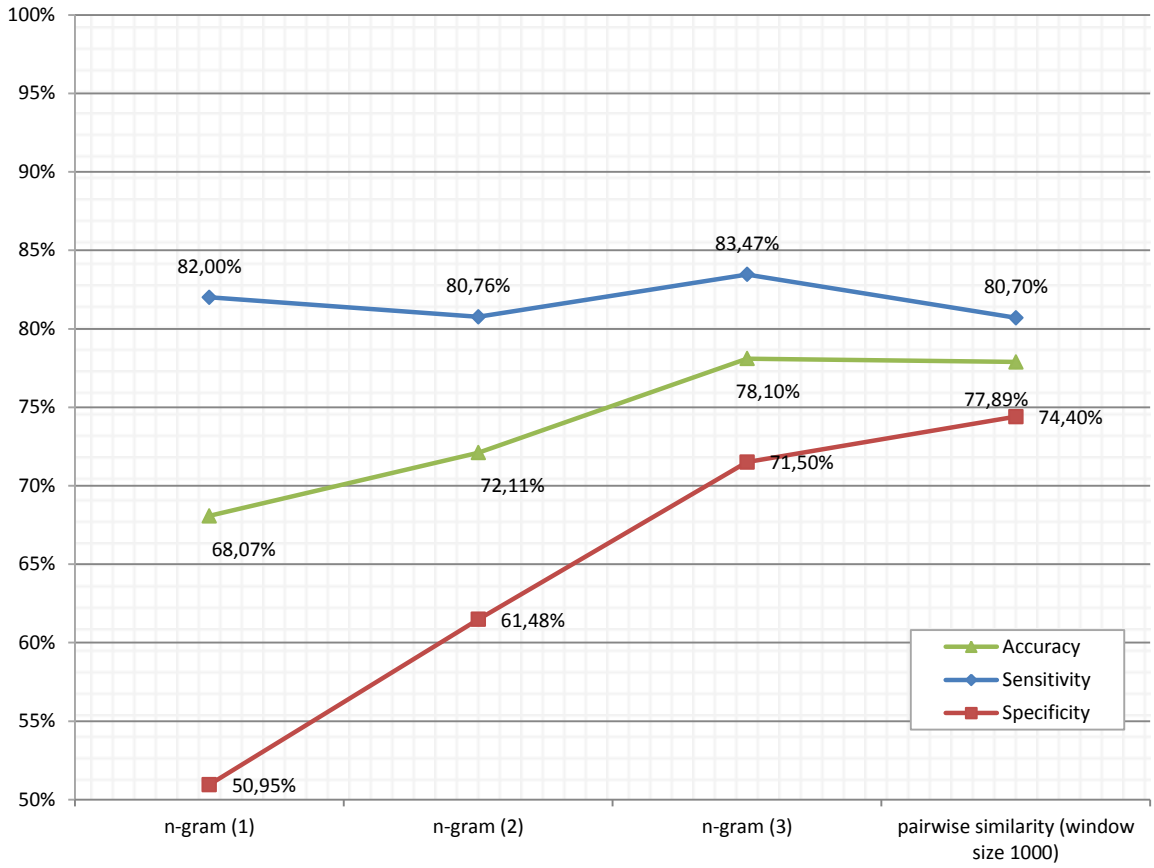


Figura 26 – Comparação dos resultados obtidos com *n*-grams com os apresentados por Zaki, Lazarova-Molnar et al. (2009) *pairwise similarity*.

CAPÍTULO 6. PLATAFORMA BIOINK SEARCH

Aplicando a metodologia de classificação e extração de características apresentada, procedeu-se ao desenvolvimento de uma plataforma web que permita de forma simples e intuitiva a deteção de peptidases. Esta aplicação está disponível no endereço www.bioink.org/bioinksearch. Para o seu desenvolvimento optou-se por uma arquitetura tradicional em camadas (como o apresentado na Figura 27).

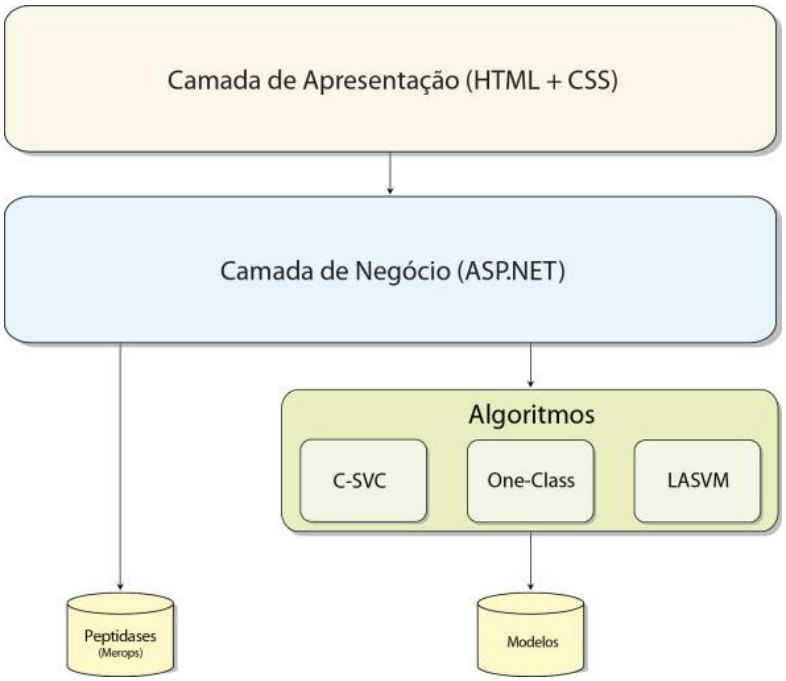


Figura 27 - Arquitetura da plataforma BioinkSearch.

A camada de apresentação revela-se uma parte essencial da plataforma, pois para os utilizadores o interface é o próprio sistema, não tendo qualquer perceção de toda a complexidade deste.

Assim, todo o sucesso da aplicação depende da qualidade do interface com o utilizador, já que se estes não forem capazes de interagir de uma forma fácil e intuitiva com a plataforma toda a utilização será prejudicada.

Deste modo, foi dada alguma importância à qualidade do interface com o utilizador, de forma a manter a consistência em toda a plataforma e ser o mais simples e intuitiva possível (Figura 28).

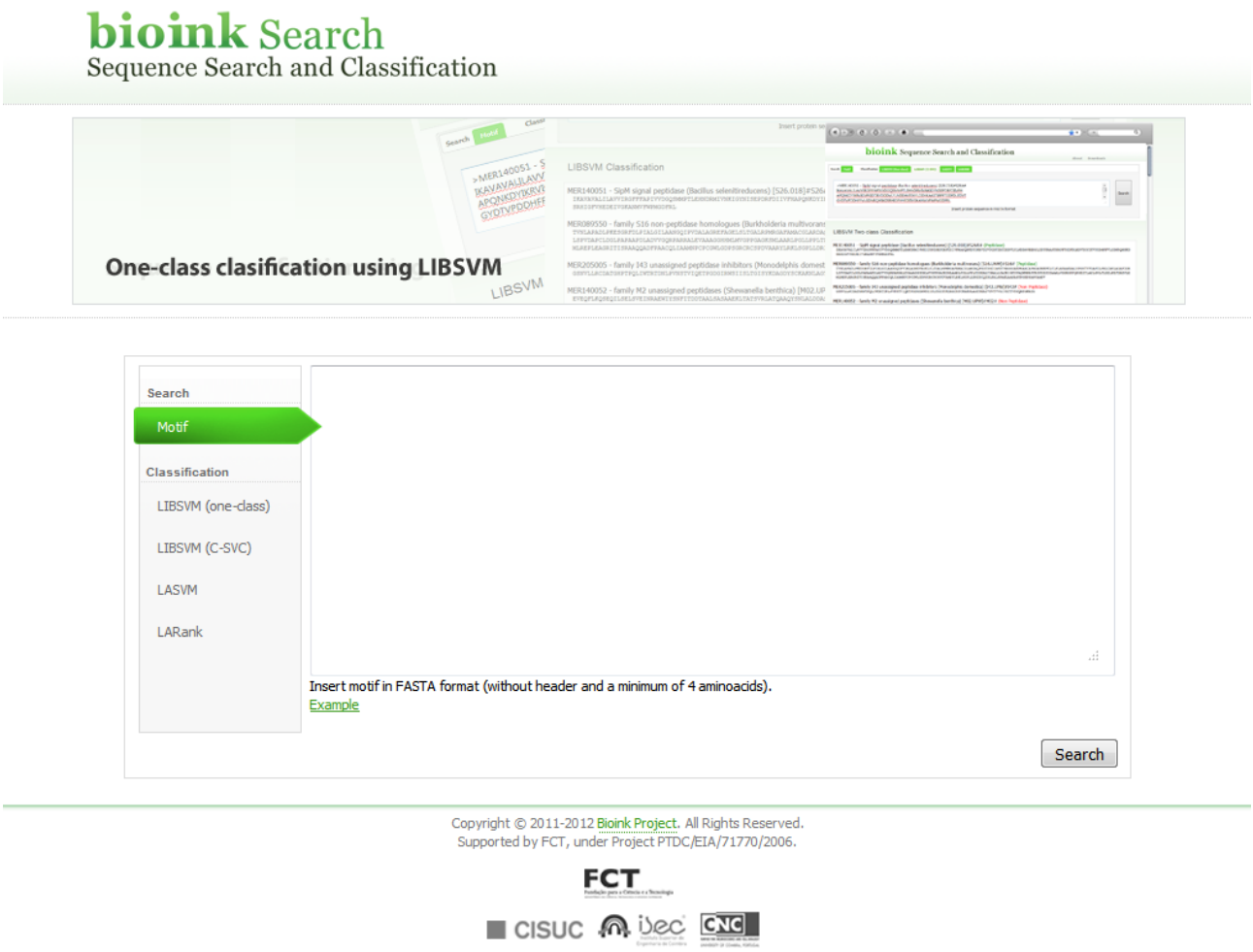


Figura 28 – Interface gráfico da plataforma Bioink Search.

Por sua vez, a camada de negócio encapsula toda a lógica da plataforma, sendo exposta à camada de apresentação para que o utilizador possa interagir com a mesma.

Para além disto dispõe de uma camada de algoritmos, de forma a facilitar a inclusão de novos algoritmos e novos modelos.

Nesta plataforma são disponibilizados três algoritmos distintos, o C-SVC, One-Class e LASVM. Os modelos usados por cada um desses algoritmos são aqueles que proporcionaram os melhores resultados.

Os utilizadores podem submeter uma sequência de aminoácidos de uma proteína no formato Fasta, de forma a aferir se é ou não peptidase. Nas Figura 29 e Figura 30 é apresentado o interface gráfico para a deteção de peptidases com os algoritmos One-Class e C-SVC respetivamente.

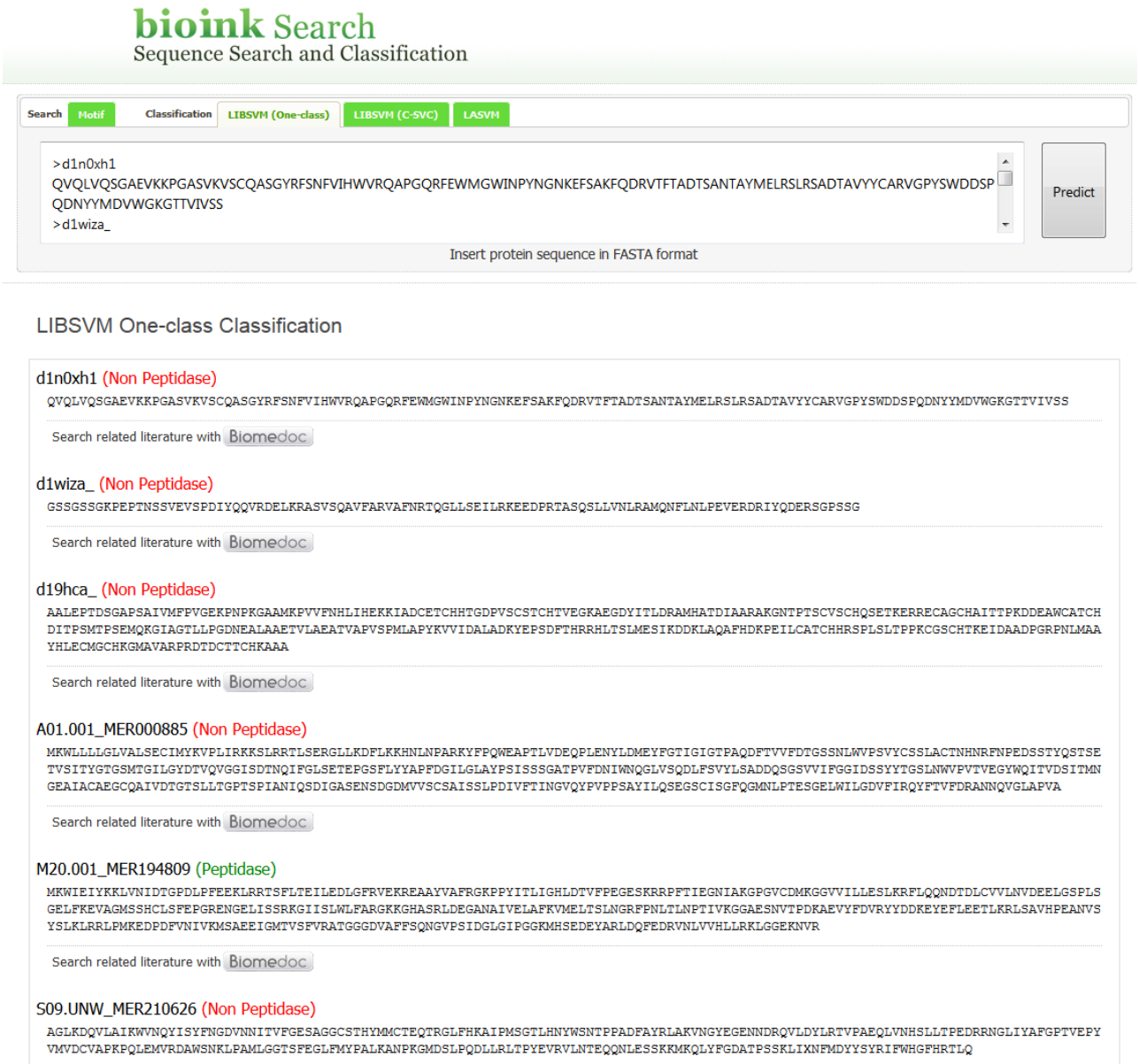


Figura 29 – Interface de deteção de peptidases com o algoritmo One-Class.

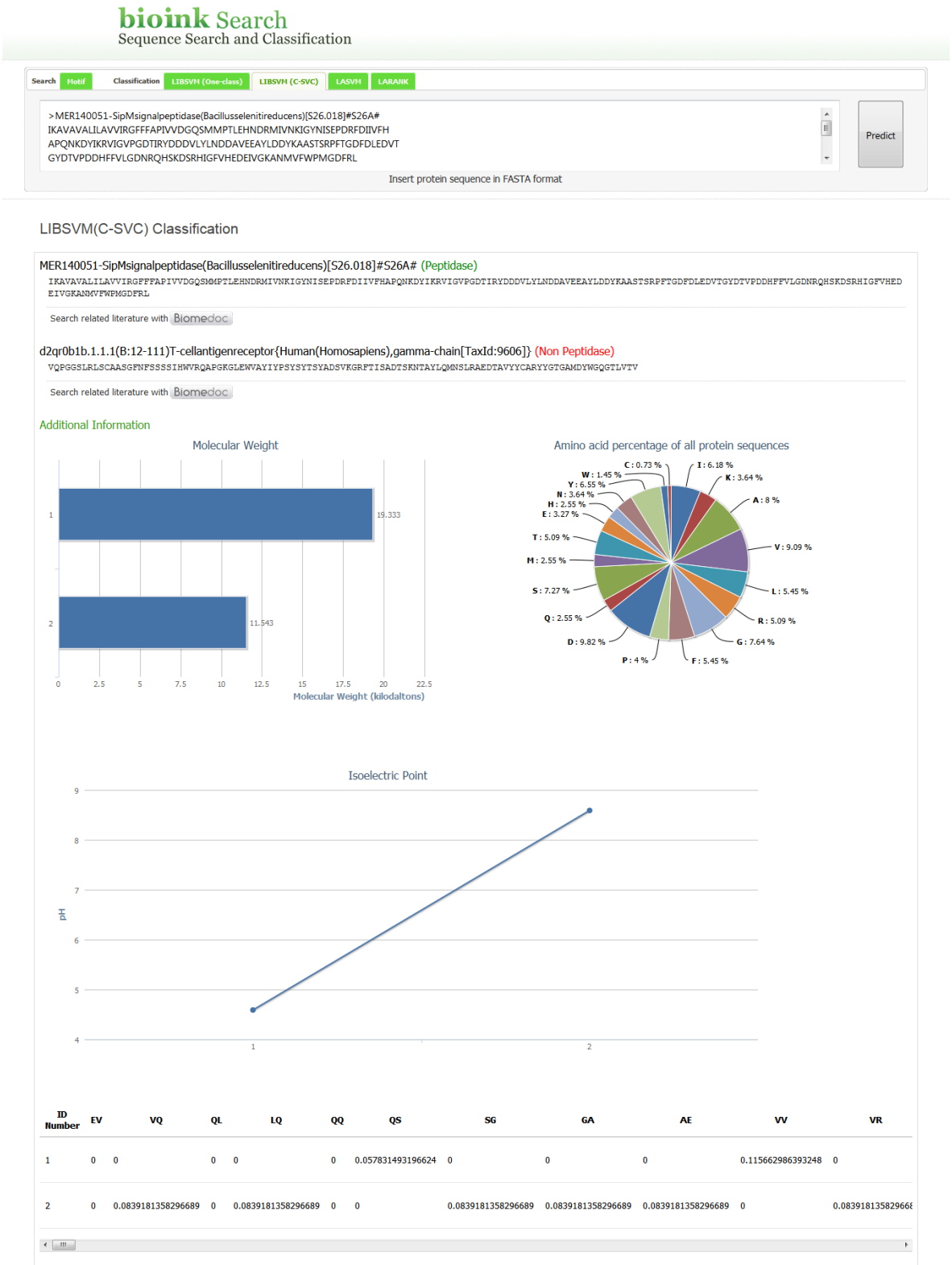


Figura 30 – Interface de deteção de peptidases com o algoritmo C-SVC.

Além das funcionalidades de classificação, foi ainda introduzida a pesquisa de Motifs, permitindo procurar em todas as proteínas presentes no *MEROPS* 8.5 padrões de aminoácidos.

Os Motifs correspondem a uma sequência particular de aminoácidos que é característica de uma funcionalidade biológica específica. Na Figura 31 é apresentado o resultado de uma pesquisa.

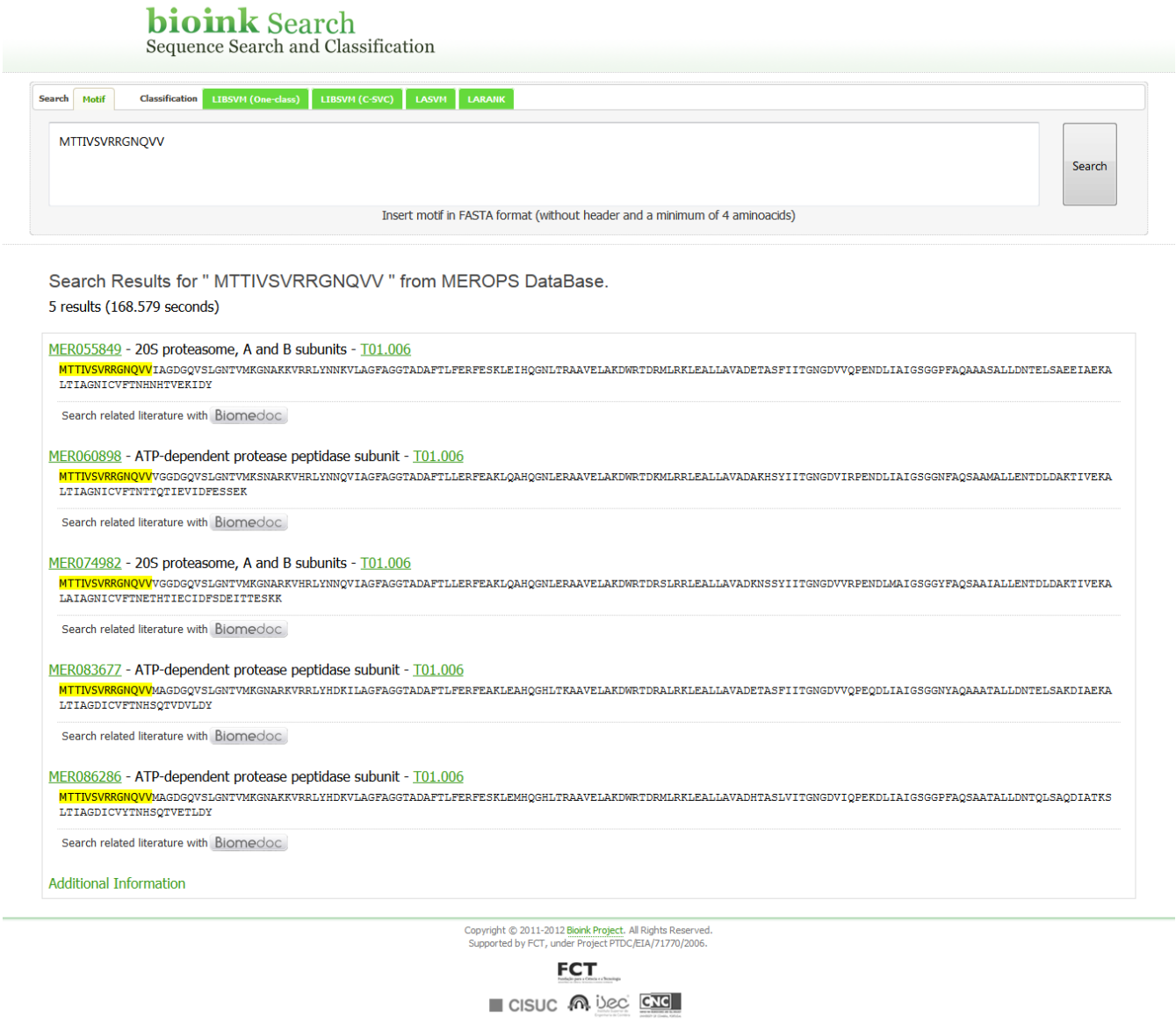


Figura 31 – Interface para pesquisa de Motifs.

No decorrer do desenvolvimento da plataforma web recorreu-se à criação de testes unitários por forma a validar o desenvolvimento (Figura 32), para isso recorreu-se à framework de testes NUnit[13].

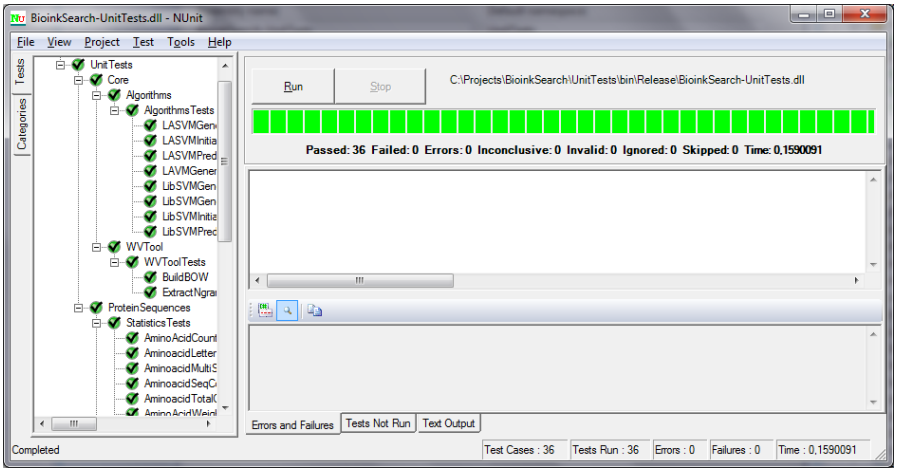


Figura 32 – Conjunto de testes unitários.

[13] NUnit : http://www.nunit.org/

CAPÍTULO 7. CONCLUSÕES E PERSPETIVAS FUTURAS

7.1. Conclusões

Ao longo desta dissertação foram aplicadas e avaliadas técnicas baseadas em *text mining* para extração de características a partir do nível primário das proteínas (sequência linear de aminoácidos). Procedeu-se à aplicação e avaliação desta metodologia em dois casos de estudo, a deteção de peptidases e a identificação de interações entre proteínas, recorrendo a algoritmos de classificação supervisionada, o LIBSVM e o LASVM.

Assim, face aos resultados obtidos e às comparações efetuadas é possível verificar que a abordagem baseada em *text mining* conjugada com os algoritmos mencionados permitem obter resultados promissores.

Particularizando, no estudo das características extraídas com as técnicas de *text mining*, verificou-se que a utilização de *n*-grams maiores que três não são suficientemente discriminativos. Este facto deve-se à forma rígida de efetuar a extração dos *n*-grams, já que só a sequência exata de aminoácidos é contabilizada.

Relativamente aos resultados do caso de estudo da deteção de peptidases, o algoritmo C-SVC com o kernel RBF foi aquele em que se conseguiram melhores resultados, obtendo aproximadamente 99% de f-measure com a combinação de *n*-grams de 1 e 2. Por sua vez, o algoritmo One-Class apresentou um f-measure de 92.70% (com *n*-grams de 3), apesar do resultado obtido ficar aquém. É de ter em consideração que este algoritmo construiu um modelo discriminativo utilizando apenas exemplos positivos. Este algoritmo pode ser usado como uma solução de compromisso quando o conjunto de dados catalogados não é balanceado ou quando só estão disponíveis exemplos de uma classe.

Já o algoritmo incremental LASVM, apesar de também não melhorar os resultados obtidos com o C-SVC, apresenta resultados bastante próximos conseguindo um f-measure de 98.40% com *n*-grams de 2. Apesar de não demonstrar ganhos ao nível dos acertos na classificação, apresenta outros benefícios bastante notórios ao nível da complexidade dos modelos gerados e do tempo de treino, tornando-se ainda mais expressivo quando a complexidade dos dados de treino aumenta. Assim, através da utilização do algoritmo

LASVM é possível tratar uma maior quantidade de dados permitindo assim aplicar a casos de estudo de maior dimensão.

Através do caso de estudo da identificação de interações entre proteínas, foi possível constatar que a abordagem apresentada consegue obter resultados similares aos apresentados na literatura, demostrando-se que a metodologia usada, apesar de simples, consegue obter resultados promissores.

Assim, os resultados obtidos e aqui apresentados seguindo a abordagem proposta são positivos e encorajadores. Com isto, conclui-se que estas técnicas podem ser alargadas a outras áreas de aplicação e usadas para auxiliar a pesquisa académica.

Relativamente ao trabalho desenvolvido, na nossa ótica, os objetivos inicialmente definidos para este trabalho foram atingidos, tendo sido estudadas e implementadas abordagens para classificação de sequências de proteínas baseadas em técnicas de *text mining* utilizando vários algoritmos de classificação.

7.2. Perspectivas Futuras

Sendo que a necessidade de classificação de dados biológicos e o desenvolvimento de algoritmos de classificação se encontram em constante evolução, a conclusão deste trabalho permite a consciencialização e orientação para o desenvolvimento de futuras investigações. Assim, refletindo um pouco sobre a problemática apresentada, enumeram-se alguns aspetos que merecem a atenção para elaboração de trabalhos futuros:

- Avaliar a introdução de novos métodos de extração de características e estender o método apresentado neste trabalho, tornando-o menos rígido, já que pequenas variações ao nível primário poderão não levar à alteração da função da proteína.
- Explorar e avaliar outros algoritmos de classificação tais como *deep learning (Lodhi 2012).*
- Explorar a aplicação de técnicas de computação evolucionária, tais como Particle Swarm Optimization (PSO) na pesquisa de parâmetros e escolha das características, de forma a automatizar o processo de treino de classificadores.
- Proceder a uma maior interligação entre as aplicações desenvolvidas para classificação de proteínas e as de pesquisa de literatura biomédica, incluindo algumas metodologias apresentadas por Correia, Campos et al. (2010) e Oliveira, Correia et al. (2011), tendo como objetivo não só auxiliar na descoberta de novo conhecimento mas também na validação dos resultados das classificações.

REFERÊNCIAS BIBLIOGRÁFICAS

Beynon, R. J. and J. S. Bond (1989). Proteolytic enzymes: a practical approach, IRL Press at Oxford University Press.

Bordes, A., S. Ertekin, et al. (2005). "Fast Kernel Classifiers with Online and Active Learning." J. Mach. Learn. Res. **6**: 1579-1619.

Caridade, F. C. G. B. (2010). Sistema Inteligente para Extracção de Características e Classificação de Proteínas. Tese de Mestrado, Instituto Superior de Engenharia de Coimbra.

Chang, C.-C. and C.-J. Lin (2011). "LIBSVM: A library for support vector machines." ACM Trans. Intell. Syst. Technol. **2**(3): 1-27.

Chen, X. W. and M. Liu (2005). "Prediction of protein–protein interactions using random decision forest framework." Bioinformatics **21**(24): 4394-4400.

Cheng, B. Y. M., J. G. Carbonell, et al. (2005). "Protein classification based on text document classification techniques." Proteins: Structure, Function, and Bioinformatics **58**(4): 955-970.

Chih-Wei, H., C. Chih-Chung, et al. (2003). "A practical guide to support vector classification." Tech. rep.(Department of Computer Science, National Taiwan University).

Cochrane, G., R. Akhtar, et al. (2009). "Petabyte-scale innovations at the European Nucleotide Archive." Nucleic Acids Res **37**(Database issue): D19-25.

Consortium, U. (2012). "Reorganizing the protein space at the Universal Protein Resource (UniProt)." Database issue(1362-4962 (Electronic)).

Correia, D., D. Campos, et al. (2010). A Platform for Intelligent Search and Classification of Biomedical Literature. Workshop on Applications of Computational Intelligence

Cortes, C. and V. Vapnik (1995). "Support-vector networks." Machine Learning **20**(3): 273-297.

De Las Rivas, J. and C. Fontanillo (2010). "Protein–Protein Interactions Essentials: Key Concepts to Building and Analyzing Interactome Networks." PLoS Comput Biol **6**(6): e1000807.

Deng, M., S. Mehta, et al. (2002). "Inferring domain–domain interactions from protein–protein interactions." Genome research **12**(10): 1540-1548.

Fau, B. D., I. Karsch-Mizrachi, et al. (2011). "GenBank." (1362-4962 (Electronic)).

jQuery Foundation. "jQuery UI." 2012, from http://jqueryui.com.

Frijters, J. "IKVM.NET: A JVM for the Microsoft.NET Framework." 2012, from http://www.ikvm.net.

Halpern, M. J. (1997). Bioquímica, Lidel.

Highsoft, S. "Highcharts - Interactive JavaScript charts for your webpage." 2012, from http://highcharts.com.

Holland, J. H. (1975). Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence. Ann Arbor, University of Michigan Press.

Hooper, N. (2002). Proteases in Biology and Medicine, Portland Press, Oxford.

Huang, C.-L. and C.-J. Wang (2006). "A GA-based feature selection and parameters optimizationfor support vector machines." Expert Systems with Applications **31**(2): 231-240.

Lodhi, H. (2012). "Computational biology perspective: kernel methods and deep learning." Wiley Interdisciplinary Reviews: Computational Statistics **4**(5): 455-465.

Morgado, L., C. Pereira, et al. (2010). PEPTILAB - A Computational Tool for Peptidase Data Analysis. Workshop on Applications of Computational Intelligence.

Murzin, A. G., S. E. Brenner, et al. (1995). "SCOP: a structural classification of proteins database for the investigation of sequences and structures." J Mol Biol **247**(4): 536-540.

Oliveira, J., D. Correia, et al. (2011). A Tool for Biomedical Documents Classification using Support Vector Machines. International Symposium on Applied Computing. Coimbra.

Rawlings, N. D., A. J. Barrett, et al. (2012). "MEROPS: the database of proteolytic enzymes, their substrates and inhibitors." Nucleic Acids Res **40**(D1): D343-D350.

Resig, J. "jquery: The write less, do more, javascript library." 2012, from http://jquery.com.

Salton, G., A. Wong, et al. (1975). "A vector space model for automatic indexing." Commun. ACM **18**(11): 613-620.

Salwinski, L., C. S. Miller, et al. (2004). "The database of interacting proteins: 2004 update." Nucleic Acids Res **32**(suppl 1): D449-D451.

Schölkopf, B., J. C. Platt, et al. (2001). "Estimating the support of a high-dimensional distribution." Neural Computation **13**(7): 1443-1471.

Schwikowski, B., P. Uetz, et al. (2000). "A network of protein-protein interactions in yeast." Nature biotechnology **18**(12): 1257-1261.

Thermo Scientific, I. (2010). "Thermo Scientific Protein Interaction Technical Handbook."

Tomović, A., P. Janičić, et al. (2006). "n-Gram-based classification and unsupervised hierarchical clustering of genome sequences." Computer methods and programs in biomedicine **81**(2): 137-153.

Vapnik, V. N. (1995). The nature of statistical learning theory. New York, Springer.

Vapnik, V. N. (1998). "Statistical learning theory." J. Wiley and Sons Inc. Nova York.

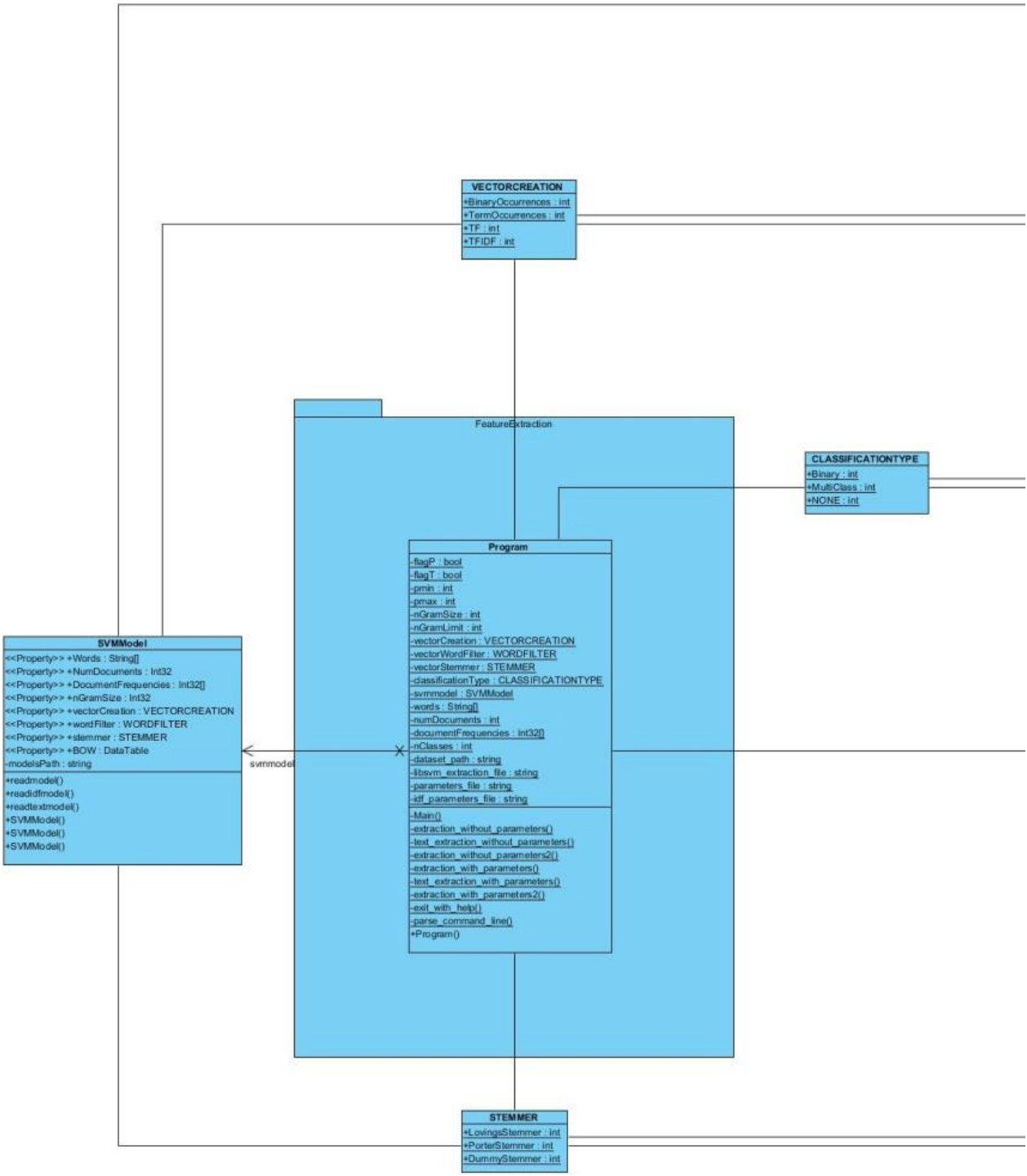
Wu, X. and V. Kumar (2009). The top ten algorithms in data mining, CRC Press.

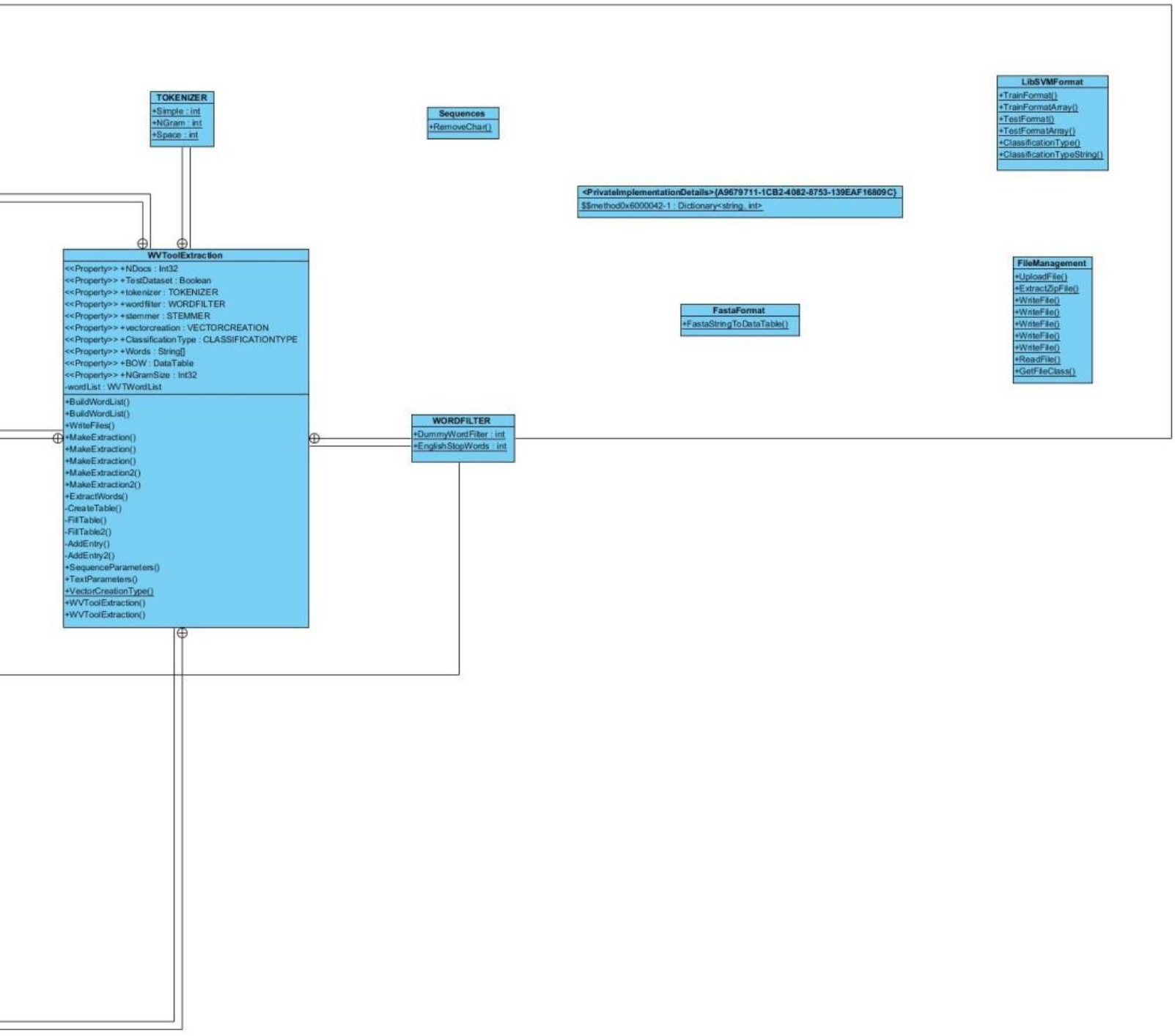
Wurst, M. (2007). "The Word Vector Tool User Guide Operator Reference Developer Tutorial."

Zaki, N., S. Lazarova-Molnar, et al. (2009). "Protein-protein interaction based on pairwise similarity." BMC Bioinformatics **10**(1): 150.

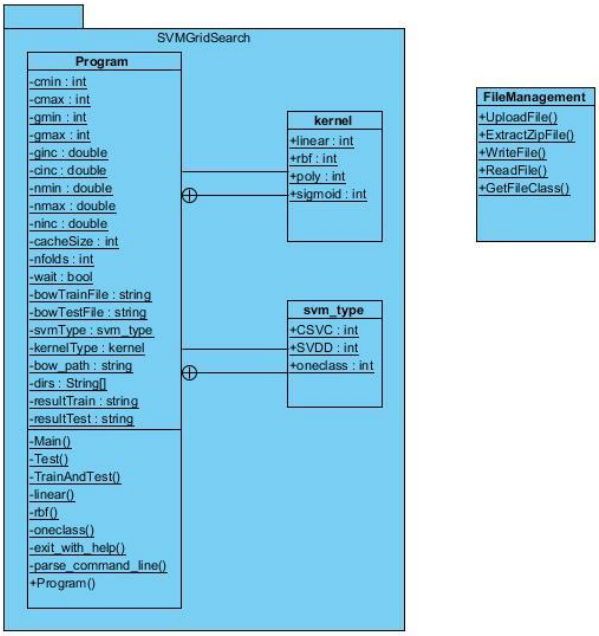
ANEXOS

Anexo A) Diagrama de classes FeaturesExtraction




TOKENIZER
+Simple : int
+NGram : int
+Space : int
Sequences
+RemoveChar()
LibSVMFormat
+TrainFormat()
+TrainFormatArray()
+TestFormat()
+TestFormatArray()
+ClassificationType()
+ClassificationTypeString()
<PrivateImplementationDetails>{A9679711-1CB2-4082-8753-139EAF16809C}
$$method0x6000042-1 : Dictionary<string, int>
WVToolExtraction
<<Property>> +NDocs : Int32
<<Property>> +TestDataset : Boolean
<<Property>> +tokenizer : TOKENIZER
<<Property>> +wordfilter : WORDFILTER
<<Property>> +stemmer : STEMMER
<<Property>> +vectorcreation : VECTORCREATION
<<Property>> +ClassificationType : CLASSIFICATIONTYPE
<<Property>> +Words : String[]
<<Property>> +BOW : DataTable
<<Property>> +NGramSize : Int32
-wordList : WVTWordList
+BuildWordList()
+BuildWordList()
+WriteFiles()
+MakeExtraction()
+MakeExtraction()
+MakeExtraction()
+MakeExtraction2()
+MakeExtraction2()
+ExtractWords()
-CreateTable()
-FillTable()
-FillTable2()
-AddEntry()
-AddEntry2()
+SequenceParameters()
+TextParameters()
+VectorCreationType()
+WVToolExtraction()
+WVToolExtraction()
FastaFormat
+FastaStringToDataTable()
FileManagement
+UploadFile()
+ExtractZipFile()
+WriteFile()
+WriteFile()
+WriteFile()
+WriteFile()
+WriteFile()
+ReadFile()
+GetFileClass()
WORDFILTER
+DummyWordFilter : int
+EnglishStopWords : int

Anexo B) Diagrama de classes SVMGridSearch



Anexo C) Resultados Preliminares Detalhados

C-SVC - Kernel RBF (BO)

	c	γ	Accuracy	Sensitivity	Specificity	Precision	Recall	F-measure
n-gram (1)	2^{6}	2^{-6}	70,38%	88,71%	52,05%	64,91%	88,71%	74,97%
n-gram (2)	2^{2}	2^{-6}	94,51%	95,01%	94,01%	94,07%	95,01%	94,53%
n-gram (3)	2^{2}	2^{-8}	94,16%	94,21%	94,11%	94,11%	94,21%	94,16%
n-gram (4)	2^{1}	2^{-9}	59,64%	100%	19,28%	55,33%	100%	71,25%

C-SVC - Kernel RBF (TO)

	c	γ	Accuracy	Sensitivity	Specificity	Precision	Recall	F-measure
n-gram (1)	2^{3}	2^{-9}	94,21%	94,11%	94,31%	94,29%	94,11%	94,20%
n-gram (2)	2^{2}	2^{-8}	95,10%	95,50%	94,71%	94,75%	95,50%	95,12%
n-gram (3)	2^{2}	2^{-8}	94,21%	94,41%	94,01%	94,03%	94,41%	94,22%
n-gram (4)	2^{-2}	2^{-6}	59,64%	100,00%	19,28%	55,33%	100,00%	71,25%

C-SVC - Kernel RBF (TF)

	c	γ	Accuracy	Sensitivity	Specificity	Precision	Recall	F-measure
n-gram (1)	2^{10}	2^{0}	90,76%	92,91%	88,61%	89,08%	92,91%	90,95%
n-gram (2)	2^{1}	2^{1}	95,05%	94,91%	95,20%	95,19%	94,91%	95,05%
n-gram (3)	2^{1}	2^{0}	94,46%	94,71%	94,21%	94,23%	94,71%	94,47%
n-gram (4)	2^{-2}	2^{1}	73,73%	69,63%	77,82%	75,84%	69,63%	72,60%

C-SVC - Kernel RBF (TF-IDF)

	c	γ	Accuracy	Sensitivity	Specificity	Precision	Recall	F-measure
n-gram (1)	2^{9}	2^{-1}	88,76%	91,91%	85,61%	86,47%	91,91%	89,10%
n-gram (2)	2^{0}	2^{1}	94,71%	94,31%	95,10%	95,07%	94,31%	94,68%
n-gram (3)	2^{1}	2^{0}	94,81%	95,10%	94,51%	94,54%	95,10%	94,82%
n-gram (4)	2^{-2}	2^{1}	73,68%	69,43%	77,92%	75,87%	69,43%	72,51%

Anexo D) Resultados C-SVC e One-Class Detalhados

C-SVC

	c	**γ**	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**Recall**	**F-Measure**
n-gram (1)	8,00	4,00	96,434%	98,793%	80,701%	97,153%	98,793%	97,966%
n-gram (2)	2,00	2,00	98,277%	99,181%	92,251%	98,842%	99,181%	99,011%
n-gram (3)	1,00	1,00	97,892%	99,303%	88,487%	98,291%	99,303%	98,794%
n-gram (1-2)	128,00	0,13	98,360%	99,290%	92,140%	98,830%	99,290%	99,059%
n-gram (1-3)	1,00	1,00	98,200%	99,491%	89,594%	98,456%	99,491%	98,971%
n-gram (2-3)	1,00	0,50	98,056%	99,181%	90,554%	98,592%	99,181%	98,886%
n-gram (1-2-3)	1,00	0,50	98,248%	99,319%	91,107%	98,675%	99,319%	98,996%

One-Class

	nu	**γ**	**Accuracy**	**Sensitivity**	**Specificity**	**Precision**	**Recall**	**F-Measure**
n-gram (1)	0,2	4	81,94%	79,89%	83,99%	83,30%	79,89%	81,56%
n-gram (2)	0,1	1	91,59%	89,85%	93,32%	93,08%	89,85%	91,44%
n-gram (3)	0,1	0,000488281	92,62%	93,73%	91,51%	91,70%	93,73%	92,70%
n-gram (1-2)	0,1	2	91,18%	89,15%	93,21%	92,92%	89,15%	91,00%
n-gram (1-3)	0,1	2,00	89,23%	86,35%	92,10%	91,62%	86,35%	88,91%
n-gram (2-3)	0,1	0,000122070	91,88%	92,44%	91,33%	91,42%	92,44%	91,93%
n-gram (1-2-3)	0,50	1	91,00%	88,56%	93,43%	93,10%	88,56%	90,77%

Anexo E) Resultados LASVM Detalhados

Resultados (LASVM)

	c	**γ**	**Accuracy**	**Sensitivity**	**Specificity**	**Recall**	**Precision**	**F-Measure**
n-gram (1)	1000	0,562341325	94,217%	97,233%	85,177%	97,233%	95,161%	96,185%
n-gram (2)	100	0,177827941	97,593%	99,078%	93,142%	99,078%	97,743%	98,406%
n-gram (3)	10	0,562341325	96,375%	99,225%	87,832%	99,225%	96,070%	97,622%
n-gram (1-2)	100	0,177827941	97,344%	99,078%	92,146%	99,078%	97,424%	98,244%
n-gram (1-2-3)	100	0,177827941	97,233%	99,594%	90,155%	99,594%	96,808%	98,181%

Tempos de Treino (LASVM)

	Finishing Step	Time (seconds)	#SV	#Train Instances	#Features
n-gram (1)	Yes	111	1532	17164	22
	No	75	1853	17164	22
n-gram (2)	Yes	81	1575	17164	473
	No	79	1815	17164	473
n-gram (2)	Yes	993	3143	17164	3378
	No	983	3190	17164	3378
n-gram (1-2)	Yes	88	1479	17164	497
	No	85	1671	17164	497
n-gram (1-2-3)	Yes	795	2026	17164	3875
	No	784	2073	17164	3875

Tempos de Treino (C-SVC - Kernel RBF)

	Time (seconds)	#SV	#Train Instances	#Features
n-gram (1)	31	1550	17164	22
n-gram (2)	180	1856	17164	473
n-gram (3)	2756	3359	17164	3378
n-gram (1-2)	144	1706	17164	497
n-gram (1-2-3)	1782	2276	17164	3875

Anexo F) Resultados C-SVC – identificação de interações entre proteínas

C-SVC - Kernel RBF (TF)

	c	γ	Accuracy	Sensitivity	Specificity	Precision	Recall	F-measure
n-gram (1)	2^{6}	2^{-6}	68,07%	82,00%	50,95%	67,27%	82,00%	73,91%
n-gram (2)	2^{2}	2^{-6}	72,11%	80,76%	61,48%	72,04%	80,76%	76,15%
n-gram (3)	2^{0}	2^{1}	78,10%	83,47%	71,50%	78,26%	83,47%	80,78%
pairwise similarity (window size 1000)			77,89%	80,70%	74,40%	-	-	-