Universidade Federal do Pampa
João Paulo de Souza Aires

Reconhecimento Automático de Expressões Temporais

Alegrete
2013

João Paulo de Souza Aires

Reconhecimento Automático de Expressões Temporais

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Fábio Natanael Kepler

Alegrete

2013

João Paulo de Souza Aires

Reconhecimento Automático de Expressões Temporais

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal do Pampa como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho de Conclusão de Curso defendido e aprovado em ..... de ..... de .....

Banca examinadora:

Prof. Dr. Fábio Natanael Kepler
Orientador
UNIPAMPA

Prof. Dr. Cleo Zanella Billa
UNIPAMPA

Prof. Dr. Sérgio Luis Sardi Mergen
UNIPAMPA

Agradecimentos

Agradeço à minha família, por ser meu apoio em toda e qualquer situação, sem ela esta monografia não teria sequer início. Agradeço também ao meu orientador, o melhor exemplo do caminho que eu quero seguir, que me auxiliou em todos os momentos, dúvidas, desatenções e principalmente, compartilhou comigo um pouco do seu vasto conhecimento. Tenho de agradecer, sem dúvida, meus amigos em geral, mas principalmente ao Holisson e ao Jader, por serem meus verdadeiros amigos dentro do curso, e serem essenciais para que eu chegasse até aqui. Muitas outras pessoas me incentivaram indiretamente para que eu seguisse no curso, através de exemplos, conselhos e também alguns sonhos compartilhados, a todos estes, meu eterno agradecimento.

Esta monografia é o resultado de toda e qualquer vivência que obtive durante minha vida, e se o mínimo detalhe fosse modificado, nada disso poderia estar acontecendo, então, no fim das contas, tenho de agradecer a todos que passaram por mim em algum momento da minha vida e fizeram desse momento possível. Obrigado!

Resumo

Neste trabalho exploramos a tarefa de reconhecimento de expressões temporais em textos de linguagem natural. Apresentamos uma visão geral das áreas de Processamento de Linguagem Natural, Extração de Informação e Reconhecimento de Entidades Mencionadas mostrando as dificuldades presentes na tarefa de reconhecimento automático de expressões que envolvem elementos temporais. Usando como abordagem métodos de Aprendizado de Máquina Supervisionado, propomos um conjunto de modelos que busca localizar as expressões temporais e classificá-las corretamente em suas categorias pré-definidas. O melhor de nossos modelos alcançou 74% de medida f utilizando nossos córpus de treinamento, o que é comparável com o estado da arte para as aplicações desenvolvidas para a língua portuguesa. Dentre os trabalhos estudados, este é o único que apresenta uma abordagem que não faz uso de regras criadas manualmente ou expressões regulares.

Palavras-chave: Reconhecimento de Expressões Temporais. Processamento de Linguagem Natural. Aprendizado de Máquina Supervisionado.

Abstract

In this work, we explore the task of identifying and classifying temporal expressions in natural language text. We present an overview of the areas of Natural Language Processing, Information Extraction, and Named Entity Recognition, and show the difficulties inherited to the task of automatically recognizing expressions involving temporal elements. By using Machine Learning methods with a supervised learning approach, we propose a set of models that try to locate temporal expressions and correctly classify them into predefined categories. The best model achieves 74% of f-measure in our corpus, which is par with the state of the art for Portuguese. However, it is the only approach we know of that does not rely on manually created rules or regular expressions.

Key-words: Extraction Temporal Expressions. Natural Language Processing. Machine Learning.

Lista de ilustrações

Lista de tabelas

Lista de siglas

AM Aprendizado de Máquina

EI Extração de Informação

ET Expressão temporal

HMM Hidden Markov Models

NLTK Natural Laguage Toolkit

PLN Processamento de Linguagem Natural

POS Part-Of-Speech

REM Reconhecimento de Entidade Mencionada

RET Reconhecimento de Expressão temporal

TCC Trabalho de Conclusão de Curso

Sumário

1 Introdução

O Processamento de Linguagem Natural (PLN) é uma área da Ciência da Computação responsável pela análise e tratamento de texto, tendo como principal objetivo alcançar um maior entendimento, por parte do computador, da linguagem utilizada pelos humanos (CHOWDHURY, 2003). Existem diversas técnicas e abordagens para realizar o tratamento de texto bem como seus limites na função devido às complexidades de lidar com as ambiguidades existentes na linguagem natural.

Dentre as diversas tarefas que fazem parte do PLN está o Reconhecimento de Entidades Mencionadas (REM), responsável pela localização e classificação dos elementos de um texto, sendo importante para tarefas ligadas a Extração de Informação, sistemas de Pergunta Resposta e Sumarização Automática (FILHO, 2011). Dentro do contexto dessas tarefas, surge a necessidade de haver uma ordenação dos dados classificados. Para isso, faz-se necessária a classificação de Expressões Temporais, que são expressões que transmitem a ideia de tempo, como data, frequência, intervalo e duração.

A utilização do Reconhecimento de Expressões Temporais (RET) se dá de várias maneiras. Existe uma grande necessidade de obter as referências temporais contidas em um texto a fim de produzir uma melhor associação dos eventos nele descritos. Desta maneira, perante algumas tarefas, a utilização dessa classificação torna-se de suma importância, como é o caso da sumarização automática multi documento, onde é gerado um resumo de vários documentos que descrevem um mesmo tema. Desta forma, a localização e classificação de expressões temporais é utilizada para que nesse documento resultante haja a sequência correta dos eventos pertencentes ao texto, permitindo ao usuário o melhor entendimento do seu conteúdo.

1.1 Objetivos

A tarefa de reconhecimento de expressões temporais apresenta diversos desafios para a área de linguagem natural, juntamente com uma grande diversidade de abordagens encontradas para sua resolução. Este trabalho tem como objetivo principal a criação de modelos computacionais que realizem o reconhecimento de expressões temporais. Utilizamos como abordagem algoritmos de aprendizado de máquina, além de ter como base os padrões de marcação já existentes, como o internacional TIMEX2/TimeML (ver Subseção 2.3.2) e as diretivas desenvolvidas pelo Segundo HAREM (ver Subseção 2.3.3). Como resultado, construímos um sistema que gera uma marcação para os elementos temporais

de um texto e classifica-os dentro dos diferentes tipos existentes em seu contexto.

1.2 Estrutura deste Trabalho

No segundo capítulo desta monografia abordamos, primeiramente, os padrões de marcação existentes para Expressões Temporais, seguido dos conceitos da tarefa de REM. Na sequência, conceituamos Expressões Temporais e as diferentes formas das quais ela pode ser reconhecida. São tratados os conceitos sobre a tarefa de RET e apresentadas algumas abordagens para sua resolução. Por fim, apresentamos alguns trabalhos relacionados e definimos a abordagem utilizada para a realização deste trabalho.

No terceiro capítulo especificamos o que foi utilizado para a construção dos modelos, abrangendo o conjunto de dados, os formatos e técnicas utilizadas juntamente com a forma de avaliação. O quarto capítulo traz o relato das experiências realizadas, bem como os resultados obtidos. No quinto capítulo, apresentamos algumas considerações finais e os trabalhos futuros.

2 Fundamentos Teóricos

Neste capítulo introduzimos a tarefa de REM, conceitualizamos a ideia de Expressão Temporal (ET) abrangendo suas principais características bem como sua importância e utilização na construção de documentos ordenados cronologicamente. São apresentadas também algumas abordagens utilizadas para a resolução do problema de classificação, seus principais conceitos e soluções.

2.1 Reconhecimento de Entidades Mencionadas

Na área de Processamento de Linguagem Natural, a Extração de Informação (EI) é a tarefa responsável pela obtenção de informação estruturada proveniente de um documento originalmente não estruturado. É através dela que torna-se possível o processamento automático dos documentos por computadores (NOBATA; SEKINE, 1998). Entre as diferentes tarefas de EI está o REM, que possui papel importante pois é responsável pela localização e classificação dos elementos contidos no texto, dividindo-os em categorias (LOUREIRO, 2007). Para realizar este reconhecimento, as entidades mencionadas são divididas em conjuntos que possuem elementos de mesma significância.

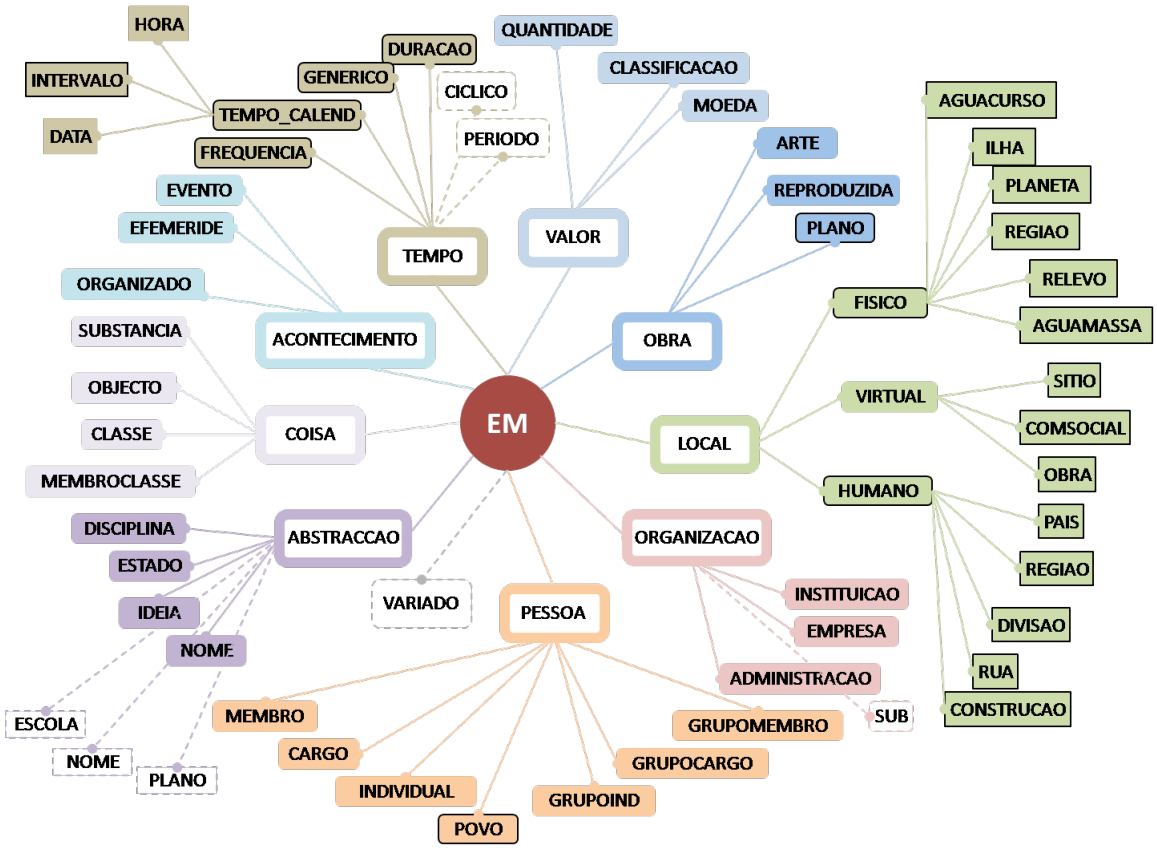
A Figura 1 ilustra a Árvore de Categorias presente no Segundo HAREM, onde é possível compreender a complexidade da tarefa de REM. A estrutura é dividida em nove categorias principais e uma (VARIADO) criada para o Primeiro HAREM. Dentro de cada categoria estão distribuídos seus devidos tipos e um nível abaixo seus subtipos. Um exemplo de sua aplicação em uma sentença pode ser dado como:

Exemplo 1 – Reconhecimento de Entidade Mencionada.

```
<EM CATEG=‘‘PESSOA’’ >Jorge</EM> trabalha desde
<EM CATEG=‘‘TEMPO’’>2006</EM> na
<EM CATEG=‘‘ORGANIZACAO’’>Universidade Federal do Pampa</EM>.
```

No Exemplo 1, a anotação foi feita através de *tags* <EM>, que identificam a palavra (entidade mencionada) por elas cercada e foi utilizado o atributo CATEG para definir a categoria do elemento identificado. O REM é útil em tarefas como sumarização automática, onde é importante o conhecimento da categoria dos elementos no texto. Isso permite a associação entre eles. Neste trabalho consideramos apenas o conceito de Entidade Mencionada, que é, segundo o REPositório para reconhecimento de ENtidades

Figura 1 – Organização da Árvore de Categorias no Segundo HAREM (CARVALHO et al., 2008)



Nomeadas (REPENTINO, 2005), "uma entidade referenciada num determinado contexto, podendo assim assumir papéis semânticos diferentes em função desse mesmo contexto.", diferentemente do conceito de Entidades Nomeadas, que é:

> palavras da classe de substantivos próprios. Definem nomes para lugares, pessoas, organizações, acontecimentos, coisas, obras, etc. É interessante observar que expressões numéricas como quantidades, valores, moeda, etc., também são consideradas uma EN, assim como expressões de tempo do tipo data, intervalos de datas, hora, período. (AMANCIO, 2009)

2.2 Expressão Temporal

A Expressão Temporal (ET) é um atributo da linguagem que transmite a ideia de tempo, uma grandeza que pode ser representada através de datas, horas, entre outras formas de expressão (MOTA et al., 2008). A ideia de tempo pode ser considerada como algo constante e que nos permite demarcar uma localização temporal, podendo ser referente a um momento descrito no calendário ou uma definição de quantidade. As expressões temporais são responsáveis por transmitir a informação sobre o tempo no qual ocorre uma situação descrita, permitindo a definição entre tempo passado, presente ou futuro. Seu uso é importante em textos como notícias, histórico hospitalar e documentos oficiais onde é necessário situar o leitor do momento no qual ocorre o que está descrito.

2.2.1 Data

Data é um tipo de ET que possui duas formas de apresentação, datas absolutas e datas referenciais. Através dela obtemos uma localização temporal mais específica, sendo possível deduzir o espaço de tempo entre o momento atual e o descrito.

2.2.1.1 Datas Absolutas

Datas absolutas são ETs que apresentam a informação completa de uma data, contendo o dia, mês e ano, podendo a mesma ser encontrada através de um calendário. Alguns exemplos de sua representação podem ser:

```
Aconteceu em 04 de setembro de 2001.
```

```
Aconteceu em 04/09/01.
```

```
Aconteceu em 04/09/2001.
```

2.2.1.2 Datas Referenciais

Dentro do conceito de data, estão as expressões que necessitam de um ponto de referência para que seja possível descobrir sua localização temporal. Elas são chamadas de datas referenciais e possuem dois modos de apresentação. As datas que fazem referência ao momento em que foi escrito e as datas que referem-se a um momento descrito no texto (BAPTISTA, 2008). Abaixo são apresentados dois exemplos de suas ocorrências de forma respectiva.

Exemplo 2 – Datas Referenciais.

```
O termo de compromisso foi assinado na tarde de ontem.
```

Exemplo 3 – Datas Referenciais.

```
Foi encontrado dois dias depois.
```

Nos exemplos apresentados, temos a necessidade de um ponto referencial no tempo para que possamos calcular o momento a qual a sentença se refere. No Exemplo 2, se tomarmos uma data referencial, como por exemplo, *18 de dezembro de 2004*, como a data em que foi escrita a sentença, podemos concluir que a expressão se refere ao dia *17 de*

```
O garoto desapareceu no dia 24 de julho de 2001 e
foi encontrado dois dias depois.
```

dezembro de 2004. Da mesma forma, podemos obter uma data contida no texto para o Exemplo 3, como no Exemplo acima.

Assim, é possível compreender que a expressão refere-se ao dia *26 de julho de 2001*. Datas referenciais podem, também, referenciar apenas datas incompletas, que possuam dois parâmetros dentre dia, mês e ano, sendo necessário, da mesma forma que os casos anteriores, a presença de uma data referencial para obter sua posição temporal.

2.2.2 Hora

A hora é um tipo de expressão temporal que pode ser categorizada como data, porém, faz referência a uma unidade inferior a dia. Suas definições permitem uma descrição mais precisa do momento de ocorrência. São reconhecidas como hora expressões como:

```
Chegaram de viagem por volta das 15hs da tarde.
```

```
O encontro ocorrereu antes das 20:30hs.
```

```
Hoje o sol raiou depois das 6 horas.
```

2.2.3 Intervalo

As expressões temporais reconhecidas como intervalo caracterizam espaços de tempo definidos por um início e um fim, e são normalmente delimitadas por outro tipo de expressão temporal, como datas e horas. Como aparecem nos exemplos abaixo:

```
Trabalho nesta empresa das 07:30 as 13:30.
```

```
Estive na praia entre 05 de janeiro a 02 de fevereiro.
```

2.2.4 Duração

As expressões temporais referentes a duração exprimem a ideia de tempo contínuo, atribuindo uma quantidade de tempo a uma determinada ocorrência. São exemplos deste tipo de expressão temporal sentenças como:

`Choveu a noite inteira.`

`Morou por dois anos na Europa.`

2.2.5 Frequência

A frequência é um tipo de ET que designa o número de vezes em que um evento acontece em uma determinada faixa de tempo. Sua ocorrência se dá como nos exemplos abaixo, onde podemos observar a presença de advérbios como no Exemplo 5, assim como a utilização de quantificadores no Exemplo 6.

Exemplo 4 – Frequêcia.

`Estou frequentando a academia diariamente.`

Exemplo 5 – Frequêcia.

`Viajo a cada dez dias.`

Exemplo 6 – Frequêcia.

`Vou ao dentista quatro vezes ao ano.`

2.2.6 Expressões Temporais Genérica

As ETs genéricas são expressões que não exprimem a ideia de uma data explicitamente, porém, possuem valor temporal que permite uma localização superficial no tempo ocorrido. Possuem esta classificação sentenças como:

`Viajaremos em Agosto.`

`Prefiro dormir no Inverno.`

A partir dos exemplos é possível concluir a falta de detalhamento em expressões deste tipo, isto porque elas não apresentam uma referência temporal consistente, porém, através delas podemos nos situar referente à época a qual condizem.

2.3 Anotação Temporal

Como o objetivo deste trabalho é realizar a anotação das expressões temporais, de forma que possamos localizá-las e classificá-las, é necessário conhecer as diferentes possibilidades de anotação existentes. Nas seções seguintes abordamos essas diferentes formas encontradas na literatura.

2.3.1 ISO8601

O ISO 8601 (ISO 8601 : 1998, "*Data elements and interchange formats - Information interchange - Representation of dates and times*") é um padrão internacional criado para a representação de datas e horas. Este padrão descreve diversos formatos para estes elementos, desde os mais básicos, sem pontuação, até os mais completos, com pontuação (WOLF; WICKSTEED, 1998).

Os formatos são apresentados com diferentes granularidades, podendo aparecer em separado. A Tabela 1 apresenta de forma superficial os diferentes formatos definidos pelo padrão.

Tabela 1 – Padrões ISO 8601.

Elemento	**Formato ISO 8601**	**Exemplo**
Ano	AAAA	1982
Ano+Mês	AAAA-MM	1982-03
Ano+Mês+Dia	AAAA-MM-DD	1982-03-22
Ano+Mês+Dia+Horas+Min+Seg	AAAA-MM-DDThh:mm:ssTZD	1982-03-22T01:44:47+03:00

Nos formatos encontramos a presença de um "T" indicando o início de uma definição de hora, e no fim a utilização do TZD, do inglês *time zone designator* (designador de fuso horário). Este último é definido através do formato +hh:mm, para fusos à direita do meridiano de *Greenwich*, -hh:mm, para fusos à esquerda ou Z.

2.3.2 TIMEX2

A forma de anotação TIMEX2 foi desenvolvida em 2000 pelo programa *Translingual Information Detection, Extraction and Summarization* (TIDES) e tem como base, para a interpretação de datas e horas, o padrão ISO8601 supracitado (TIMEX2,). Voltada para auxiliar em tarefas de reconhecimento, o padrão é moldado através de *tags* XML (*eXtensible Markup Language*). Essas *tags* são utilizadas para localizar os elementos, pois marcam o início <TIMEX2> e o fim </TIMEX2> da expressão. As *tags* podem possuir até seis atributos Tabela 2 que serão utilizados para a classificação da expressão. Sua utilização está representada no Exemplo 7.

Tabela 2 – Padrões TIMEX2.

Atributo	Descrição	Exemplo
VAL	Contém a forma normalizada de uma data ou hora na expressão.	VAL = “T01:30:29Z”
MOD	Captura os modificadores de tempo, usando valores como BEFORE (antes), MORE_THAN (mais que), START(início) e APPROX (aproximadamente).	MOD = “START”
ANCHOR_VAL	Contém uma forma normalizada de uma âncora de data ou tempo.	ANCHOR_VAL = “1982-03-22”
ANCHOR_DIR	Captura a direção ou orientação relativa entre os atributos VAL e ANCHOR_VAL, podendo ser representado através dos valores WITHIN (dentro), Starting (iniciando) e BEFORE (antes).	ANCHOR_DIR = “WITHIN”
SET	Identifica expressões que denotam conjuntos temporais.	SET = “YES”
COMMENT	Contém qualquer comentário do qual o anotador queira fazer.	COMMENT = “Comentário”

Exemplo 7 – TIMEX2

```
O jogo aconteceu <TIMEX2>duas horas antes</TIMEX2> do previsto.
```

2.3.3 Segundo HAREM

O Segundo HAREM é a segunda edição do HAREM, uma avaliação conjunta organizada pela LINGUATECA, realizada na área de reconhecimento de entidades mencionadas para a língua portuguesa (CARVALHO et al., 2008). Caracteriza uma avaliação conjunta, a utilização de um modelo avaliativo no qual diferentes grupos fazem uso para comparar suas aplicações referentes a tarefas definidas sobre uma determinada área de conhecimento. Tendo como foco a identificação e classificação das diferentes expressões encontradas nos textos, o Segundo HAREM propõe uma anotação para as entidades mencionadas. Sendo as expressões temporais um ramo das entidades mencionadas, foi criado um padrão para tratá-las. Este padrão traz uma marcação através de *tags* XML onde o atributo CATEG define a categoria da entidade mencionada reconhecida, no caso das expressões temporais a *tag* é assim representada: <EM CATEG=“TEMPO”></EM>. Além do atributo CATEG foram criados outros, como TIPO, SUBTIPO, TEMPO_REF, SENTIDO, VAL_DELTA e VAL_NORM, que podem ser melhor entendidos na Tabela 3.

Um outro atributo existente é o VAL_DELTA, ele é utilizado apenas na presença do atributo TEMPO_REF e quando este recebe os valores ENUNCIACAO ou TEXTUAL. O atributo VAL_DELTA tem como valor uma expressão que permite a compreensão da distância temporal entre o tempo do evento apresentado pela expressão temporal

Tabela 3 – Padrão Segundo HAREM.

Atributo	**Descrição**	**Exemplo**
TIPO	Único atributo obrigatório, podendo receber os seguintes valores: TEMPO_CALEND(tempo calendário), DURACAO (duração), FREQUENCIA (frequência) e GENERICO (genérico).	TIPO=“DURACAO”
SUBTIPO	Ocorrem quando o TIPO é definido como TEMPO_CALEND, podendo receber valores como: DATA, HORA e INTERVALO.	SUBTIPO=“DATA”
TEMPO_REF	Atributo utilizado em relação ao subtipo DATA, possuindo o valor ABSOLUTO quando a data for completa (dia, mês e ano), o valor TEXTUAL, quando se trata de uma data referencial, com referência textual ou ENUNCIACAO, quando se referir a uma data referencial relativa ao momento de enunciação.	TEMPO_REF=“ABSOLUTO”
SENTIDO	Atributo utilizado em relação ao atributo TEMPO_REF com os valores TEXTUAL e ENUNCIACAO, auxiliando na normalização das datas referenciais. Pode receber como valores ANTERIOR, POSTERIOR, SIMULT, ANTERIOR_OU_SIMULT e POSTERIOR_OU_SIMULT.	SENTIDO=“ANTERIOR”

e o momento de referência (HAGÉGE; BAPTISTA; MAMEDE, 2008). Na falta de uma distância temporal, o valor do atributo é omitido. A valorização do atributo acontece baseado no seguinte formato:
A<digitos>M<digitos>S<digitos>D<digitos>H<digitos>M<digitos>S<digitos>

As letras referenciam respectivamente Ano, Mês, Semana, Dia, Hora, Minuto e Segundo. Os <digitos> referenciam os números que devem ser atribuidos para representar a distância temporal da data referencial, obtendo assim o valor da expressão anotada. Um exemplo da sua utilização pode ser observado no Exemplo abaixo.

```
A viagem ocorreu <EM ID=‘‘...’’ CATEG=‘‘TEMPO’’ TIPO=‘‘TEMPO_CALEND’’
SUBTIPO=‘‘DATA’’ TEMPO_REF=‘‘TEXTUAL’’ SENTIDO=‘‘ANTERIOR’’
VAL_DELTA=‘‘A0M1S0D4H0M0S0’’>um mes e quatro dias</EM>
antes do previsto.
```

Um outro atributo conhecido como VAL_NORM, é utilizado com relação aos atributos TIPO, quando este receber o valor de DURACAO ou TEMPO_CALEND, e SUBTIPO quando for referente à DATA ou HORA. O VAL_NORM usado no contexto de DURACAO, utiliza o mesmo padrão apresentado pelo atributo VAL_DELTA, A<digitos>M<digitos>S<digitos>D<digitos>H<digitos>M<digitos>S<digitos>. No contexto de DATA, quando o TEMPO_REF for ABSOLUTO, o VAL_NORM respeita o formato:
<Era><Ano><Mes><Dia>T<Hora><Minuto>E<Estacao>LM<limite_aberto>

<Era> recebe o valor de "-" ou "+", definindo a data como antes ou depois da era de referência. <Ano> recebe 4 dígitos, indicando o ano referente, <Mes> recebe 2 dígitos, referenciando os meses de 01 até 12, <Dia> recebe até 2 dígitos, referenciando dias de 01 até 31. Ao iniciar a representação de hora, é adicionado um "T" na expressão, que é seguida de <Hora> que recebe 2 dígitos indicando a hora e <Minuto> que também recebe 2 dígitos para representar os minutos. Seguindo a expressão, é adicionado um "E" que indica que o próximo valor a ser recebido é referente a uma estação, sendo assim <Estacao> pode receber os valores: PR (primavera), VE (verão), OU (outono) e IN (inverno).

Por fim, são adicionadas as letras "LM", referenciando um limite aberto, ou seja, indica se a expressão normalizada possui um intervalo de tempo com limite anterior ou posterior em aberto. O campo <limite_aberto> pode receber o valor A, para o caso de um limite anterior em aberto, P, no caso de limite posterior em aberto e "-" quando a data não corresponde a um intervalo com um dos limites em aberto. A utilização deste atributo pode ser visualizado no Exemplo abaixo:

```
Em <EM ID=‘‘...’’ CATEG=‘‘TEMPO’’ TIPO=‘‘TEMPO_CALEND’’
SUBTIPO=‘‘DATA’’ TEMPO_REF=‘‘ABSOLUTO’’
VAL_NORM=‘‘+18320220T----E--LM-’’>20 de fevereiro de 1832</EM>
Charles Darwin visitou Fernando de Noronha.
```

O atributo VAL_NORM pode ser também utilizado em complemento do SUBTIPO HORA, utilizando a formatação mostrada anteriomente.
<Era><Ano><Mes><Dia>T<Hora><Minuto>E<Estacao>LM<limite_aberto>

Porém, algumas definições são diferenciadas, os campos <Era>, <Ano>, <Mes> e <Dia>, são preenchidos com +- - - - - - - -, pois não são referentes a hora, o mesmo acontece com <Estacao> que recebe um "–". Por fim, o campo <limite_aberto> pode corresponder aos valores D ou E, que significam respectivamente a presença de um intervalo aberto anterior ou posterior.

2.4 Reconhecimento de Expressões Temporais

Na tentativa de resolver o problema de REM, concluiu-se que, devido à extensa área que as expressões temporais abrangem, assim como sua complexidade de classificação, seria necessário dividir o problema de classificação de ET em um caso especial. Dessa forma, originou-se a tarefa de Reconhecimento de ETs, com o objetivo de localizar e classificar as expressões temporais presentes em documentos de texto, utilizando para isso as diferentes categorias que as ET apresentam.

2.4.1 Abordagens para Resolução

Como forma de obter um melhor entendimento e assim alcançar melhores resultados, foram analisados diferentes tipos de abordagens para a resolução do problema de Localização e Classificação de ET. Dentre as diferentes abordagens encontradas, estão os Métodos Simbólicos, que utilizam técnicas como Expressões Regulares e anotação manual, e o Aprendizado de Máquina que possui abordagens de Aprendizado Supervisionado e Não Supervisionado.

2.4.1.1 Métodos Simbólicos

Os Métodos Simbólicos representam boa parte das aplicações encontradas. Suas características definem programas que fazem uso de informações pré-definidas, utilizando regras em cima dessas informações para obter seus resultados (OSORIO; VIEIRA, 1999). Um exemplo deste tipo de utilização encontrado como resolução para o problema de Localização e Classificação de ETs, foi o uso de expressões regulares como mecanismo de localização. Baseado no contexto vinculado a um tipo de ET, são criadas regras para a expressão regular que irá reconhecer elementos dentro do texto a partir do padrão definido. Dessa forma, os elementos reconhecidos já pertencem a uma categoria definida previamente na construção da regra.

Existem alguns padrões de anotação desenvolvidos, como o <TIMEX2>, originalmente criado para o reconhecimento de ET da língua inglesa, e o <EM>, criado para a língua portuguesa. Ambas abordagens são representadas através de *tags* que delimitam a área na qual a ET está situada. Como forma de complementar a classificação dos elementos, as *tags* apresentam como recurso o uso de atributos para especificar categorias e subcategorias da expressão. Um exemplo do uso da anotação baseado nas definições do Segundo HAREM (CARVALHO et al., 2008), pode ser visto abaixo:

```
Choveu ate as <ET ID=‘‘...’’ TIPO=‘‘TEMPO_CALEND’’
SUBTIPO=‘‘HORA’’>17 horas</ET>
de hoje, o periodo de chuvas que ocorre
<ET ID=‘‘...’’ TIPO=‘‘DURACAO’’>entre Abril e Junho</ET>
causa diversos estragos no estado.
```

Ambas abordagens deste gênero apresentam desvantagens quando utilizadas em ambientes diferentes dos previamente programados, tornando praticamente impossível sua utilização em uma linguagem diferente, uma vez que as definições utilizadas abrangem apenas questões de um único contexto. Outra desvantagem aparece por conta da limitação que gera a utilização de Métodos Simbólicos, não há um tratamento para o caso de incertezas, valores aproximados e informações contraditórias, acabando, assim, por gerar resultados indesejados ao generalizar certas marcações, desconsiderando a presença des-

ses elementos tão presentes na realidade das aplicações sobre o contexto de Linguagem Natural (CARVALHO, 2012).

2.4.1.2 Aprendizado de Máquina

O Aprendizado de Máquina (AM) surgiu como uma área da Inteligência Artificial, tendo como objetivo central a criação de algoritmos que permitam um aprendizado automático, por parte da máquina, através da análise de dados. Com isso, torna-se possível classificar novos elementos baseados nas suas características (MONARD; BARANAUSKAS, 2003). Este processo de aprendizado faz uso do princípio da indução, que busca através de análises de conjuntos de dados obter hipóteses, mesmo que genéricas, para definir estes conjuntos.

O Aprendizado de Máquina Supervisionado é uma técnica de AM que faz uso de elementos previamente rotulados para então etiquetar elementos não rotulados. Sua execução é feita através da utilização de um conjunto de treinamento, o qual é constituído pelos elementos previamente anotados. Este conjunto é então utilizado para treinar o algoritmo de aprendizado. Também é definido um conjunto de teste, que irá servir como forma de avaliação do algoritmo treinado. Para obter valores de eficiência do algoritmo de aprendizado, existem medidas como: Precisão, Cobertura e Medida F (veja Seção 3.4).

No conjunto de abordagens que utilizam Aprendizado de Máquina Supervisionado para resolver o problema de Localização e Classificação de ET, os processos baseiam-se no uso de dados previamente anotados, neste caso, documentos que apresentem anotação de ET como base para a criação do conjunto de treinamento. Este treinamento irá permitir ao algoritmo reconhecer, baseado nas informações passadas pelos documentos marcados, características semelhantes em novos elementos.

Existem diversos algoritmos de aprendizado encontrados na literatura que podem ser usados para a tarefa, tais como: *Naive Bayes*, Máxima Entropia, *Conditional Random Fields*, entre outros.

O aprendizado de máquina não supervisionado tem como principal característica a não utilização de dados prévios, ou seja, todo cálculo realizado se baseia apenas na utilização das informações passadas como entrada (SOUTO et al., 2003). Essa técnica é normalmente utilizada quando se faz necessária a descoberta de padrões presentes em um conjunto de dados. Para que essa descoberta seja feita, são analisadas as características desses dados, buscando semelhanças e assim agrupando-os em diferentes grupos. O principal algoritmo que utiliza esta técnica é o k-médias (*k-means*), também havendo os modelos de mistura (*mixture methods*) e o agrupamento hierárquico (*hierarchical clustering*).

2.5 Trabalhos Relacionados

Visando entender os diversos desafios da tarefa de Localização e Classificação de ET, foram analisadas algumas abordagens que a implementam. São apresentadas nas subseções seguintes os diferentes tipos de abordagens encontradas.

2.5.1 PorTexTO: sistema de anotação/extração de expressões temporais

O sistema PorTexTO (*PORtuguese Temporal Expressions Tool*) (CRAVEIRO; MACEDO; MADEIRA, 2008) foi desenvolvido com o objetivo de reconhecer ETs da língua portuguesa em diferentes tipos de documentos. Participou do livro Segundo HAREM publicado em 2008, onde foi avaliado nas tarefas de identificação e classificação das ETs.

O processo do sistema se dá através da quebra do documento em frases. Essas frases passam então por uma etapa de identificação de ETs, que é realizada através do uso de expressões regulares (REGEX) geradas a partir da análise do contexto de forma manual e fazendo uma associação baseada na classificação segundo as diretivas do Segundo HAREM. O sistema é dividido em dois módulos, Processador de coocorrências e Anotador, detalhados abaixo.

O módulo Processador de coocorrências tem como objetivo a criação de expressões regulares a partir de análises da utilização de ETs. Com isso são definidos padrões para um determinado contexto de aplicação, sendo utilizado apenas quando há a necessidade da criação das mesmas ou no caso de haver expressões regulares insuficientes. Seu funcionamento tem início partindo da análise de um conjunto de palavras que possuem referência temporal, como meses do ano, dias da semana e unidades de medida temporal, para que seja então gerada uma lista dessas expressões. Após essa análise, é feita uma agregação entre as ET da lista gerada, buscando unir expressões que possuam mais de uma palavra, no mesmo local, relacionada, como nas frases: *No* ***ano*** *passado*, *No* ***ano*** *seguinte*. Desta forma será feita a agregação, gerando um padrão: *No* ***ano*** *passado* | *seguinte*. Feito isto, a lista é então organizada e através de uma análise manual são geradas as REGEX que definirão os padrões que serão utilizados pelo módulo Anotador. A Figura 2 ilustra a arquitetura do módulo.

O módulo Anotador tem como função receber documentos e padrões definidos pelo módulo Processador de coocorrências e então identificar as ETs contidas nos documentos, classificando-as e por fim realizando as anotações geradas. Seu procedimento se divide em quatro etapas sequenciais. O processamento é feito em um documento por vez, ele é então quebrado em frases e passa para a primeira etapa do processo, onde as frases serão selecionadas de forma a evitar o processamento de frases que não possuam ET. Na segunda etapa são escolhidas expressões candidatas para marcar as ET encontradas na frase, então na terceira etapa essas marcações são comparadas com os padrões definidos.

Figura 2 – Arquitetura do Módulo Processador de co-ocorrências do PorTexTO.

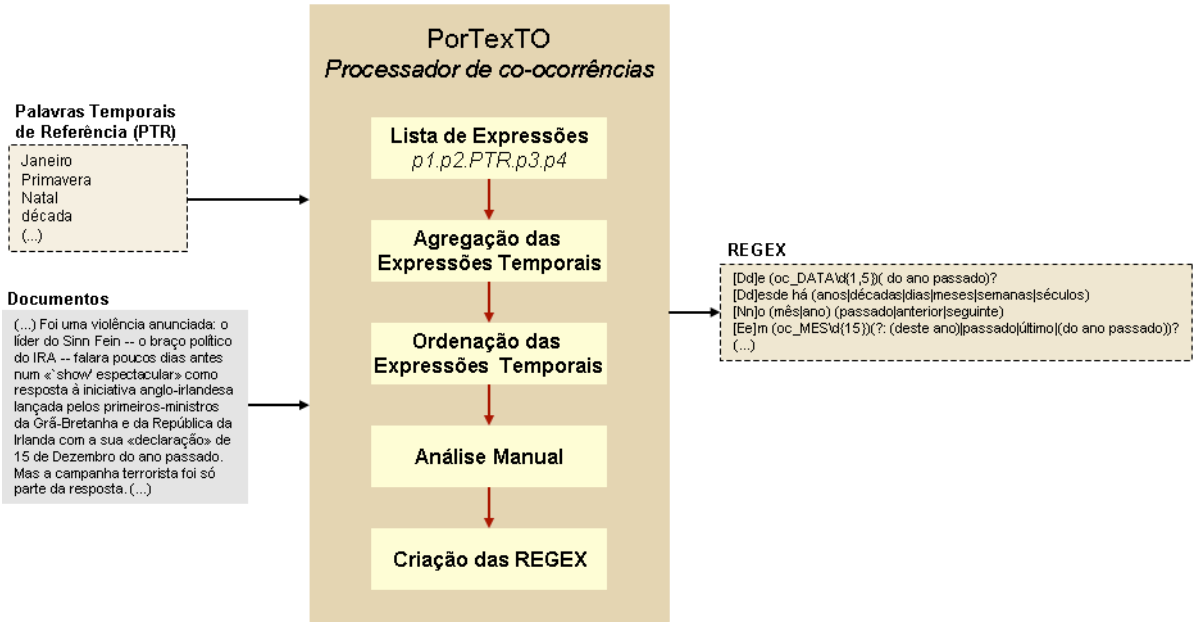
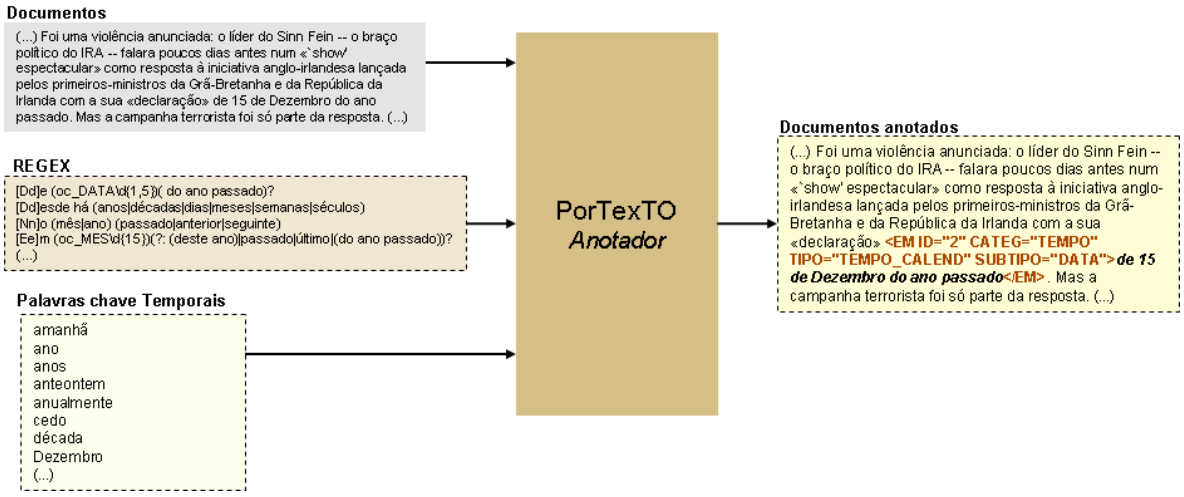


Figura 3 – Arquitetura do Módulo Anotador de co-ocorrências do PorTexTO.



Caso haja correspondência entre os elementos, é feita a anotação. Na quarta etapa as marcações colocadas na segunda etapa são substituídas pelo texto original. Na Figura 3 é possível visualizar a arquitetura do módulo.

2.5.2 CoppeTER

O sistema CoppeTER (*Coppe Temporal Expression Recognizer*) (FILHO, 2011) foi desenvolvido para uma dissertação de mestrado apresentada em 2011 ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro. O sistema apresenta como proposta uma abordagem para a tarefa de reconhecimento e normalização de ET da língua portuguesa, com um esquema de anotação que busca utilizar tanto aspectos do padrão internacional (TIMEX2/TimeML)

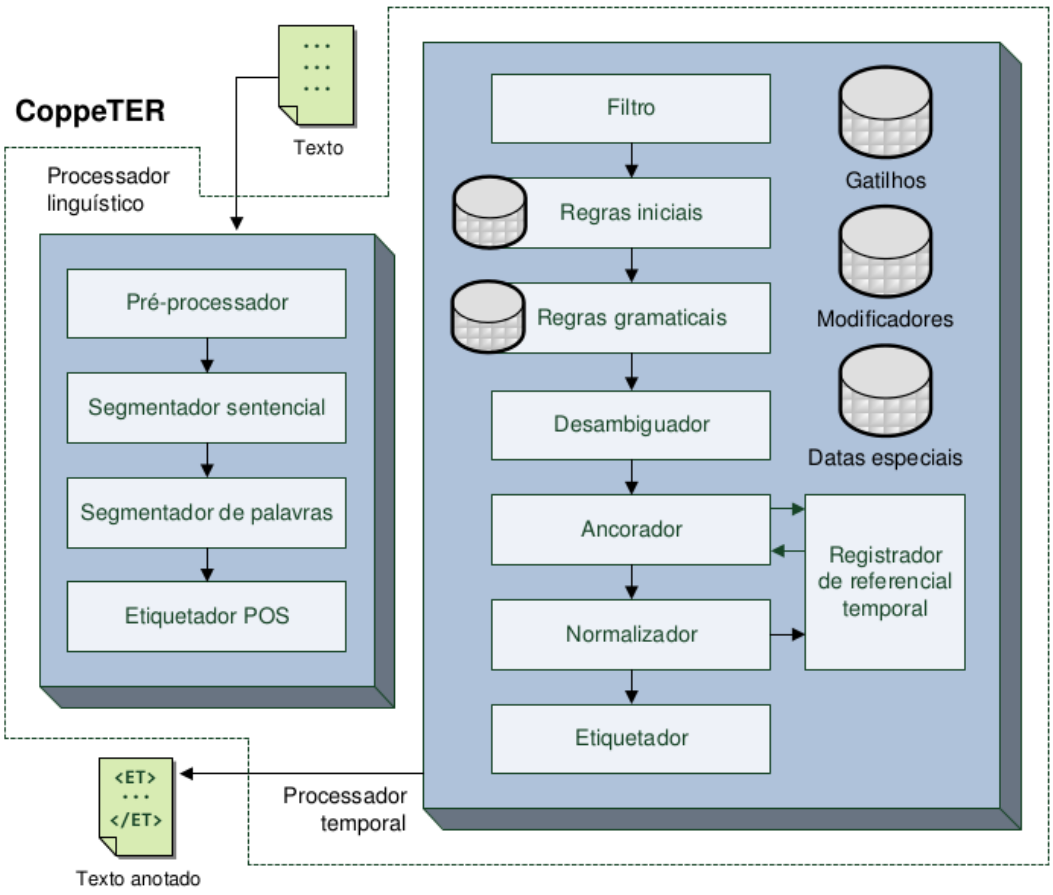
como as diretivas criadas para o Segundo HAREM. Sua implementação foi realizada utilizando a linguagem de programação Python, bem como a plataforma de processamento de Linguagem Natural, *Natural Language Toolkit* (NLTK). A arquitetura do CoppeTER é dividida em dois componentes, Processador Linguístico e Processador Temporal, como é possível ver na Figura 4.

O Processador Linguístico é o primeiro componente do sistema, ele é dividido em quatro módulos que ao final resultam num conjunto de sentenças etiquetadas. O componente recebe como entrada o texto a ser processado, que é então encaminhado ao primeiro módulo, o pré-processador, que irá preparar o texto para ser passado aos próximos módulos. O segmentador sentencial recebe o texto como entrada e gera uma lista de sentenças, as quais são passadas, uma a uma, ao segmentador de palavras, que irá retornar uma lista de *tokens* pertencentes à sentença. Por fim, o etiquetador POS irá processar os *tokens*, gerando pares contendo o *token* e sua devida etiqueta POS (veja Subseção 3.1.2). O retorno gerado pelo último módulo servirá como entrada para o segundo componente do sistema.

O segundo componente do sistema, chamado de Processador Temporal, é constituído de três dicionários e um conjunto de regras que alimentam os módulos com dados. No filtro, primeiro módulo do componente, as frases são selecionadas através de um dicionário de gatilhos, excluindo aquelas que não possuírem ao menos uma palavra com referência temporal. Na sequência, as frases passam por duas etapas de processamento sobre um conjunto de regras, sendo a primeira responsável pela identificação de expressões que possuam padrões simples e, através de um dicionário, são também reconhecidas expressões que façam referência a datas comemorativas, feriados, entre outras referências encontradas no calendário. A segunda etapa, através de regras gramaticais, faz o reconhecimento das expressões temporais contidas nas frases.

Para o caso de alguma ET apresentar ambiguidade ou for relativa, as frases passam pelos módulos desambiguador e ancorador. No módulo desambiguador, as ambiguidades são resolvidas através de um processo que utiliza aprendizado de máquina, fazendo uso de classificadores de máxima entropia. No módulo ancorador, é feita a identificação do foco temporal expressado por uma ET. Por fim, o módulo normalizador irá utilizar as informações do dicionário da expressão temporal para que sejam normalizadas as informações temporais contidas na definição de ET e então o módulo etiquetador ira formatar as *tags* e atributos pertencentes a elas.

Figura 4 – Arquitetura do sistema CoppeTER (FILHO, 2011).

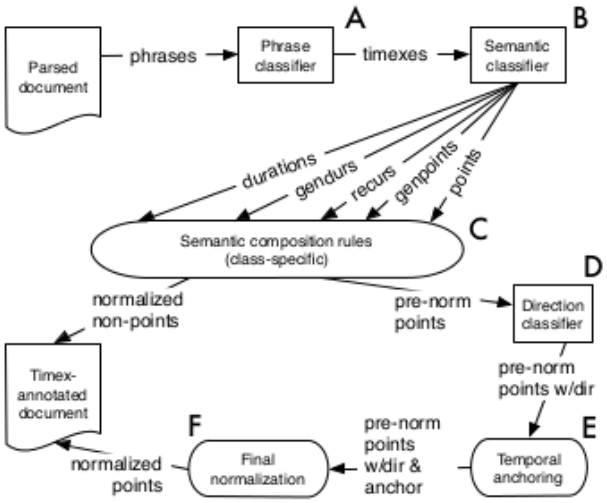


2.5.3 Uma Abordagem de Aprenidizado de Máquina em Cascata para Interpretação de Expressões Temporais

O sistema de Interpretação de Expressões Temporais (AHN; RANTWIJK; RIJKE, 2007), foi desenvolvido pela *Intelligent System Lab Amsterdam* (ISLA), Universidade de Amsterdam, com o objetivo principal de apresentar uma abordagem que utiliza aprendizado de máquina no seu processamento, evitando assim a necessidade do uso de diversas regras pré-definidas.

A arquitetura do sistema é dividida em seis módulos, como é possível observar na Figura 5. Eles estão identificados por letras de A a F. O primeiro módulo (A) é responsável pelo reconhecimento das expressões temporais num documento, fazendo uso de um *parser* (CHARNIAK, 2000). Ele busca classificar frases candidatas através de uma categorização léxica, com isso são selecionadas as frases que, por conterem ET, serão enviadas para o segundo módulo. O segundo módulo (B) é um classificador de classes semânticas, que tem como tarefa a desambiguação das ET que apresentam mais de um sentido e a classificação dos elementos. Para sua realização, são explicitadas cinco categorias: referência, duração genérica ou vaga, duração, ponto genérico ou vago (referem-se a hora, expressão de datas sem referência e referências gerais ao passado, presente ou futuro) e

Figura 5 – Arquitetura do sistema Timex Annotation. (AHN; RANTWIJK; RIJKE, 2007)



ponto (expressão de datas com referência). O terceiro módulo (C) trata da composição semântica das expressões, buscando, através da análise por regras de combinação padrão, gerar uma representação semântica que permite sua tradução em um valor normalizado. Com exceção das ET que referem-se a um ponto específico no tempo, todas as outras serão normalizadas. São também tratadas as expressões que necessitam de um ancoramento para sua normalização, normalmente presente no texto. Para o caso das expressões referenciais que possuem uma direção de relação indefinida, no quarto módulo (D) é utilizado um classificador com aprendizado de máquina para resolver estas ambiguidades. O quinto módulo (E) é responsável por definir âncoras temporais não existentes. Essas âncoras são datas presentes dentro do texto que servem de referência temporal às outras. O módulo diferencia as expressões dêiticas das anafóricas e utiliza métodos heurísticos de ancoração temporal. O módulo final (F) combina as representações semânticas com as âncoras temporais e a classe de direcionamento, gerando como resultado final o documento anotado.

2.6 Abordagem Proposta

De modo que o objetivo deste trabalho fosse realizado, buscando uma abordagem diferenciada das hoje existentes para a tarefa de reconhecimento de expressões temporais, decidimos pela utilização da técnica de aprendizado de máquina supervisionado. A sua utilização vem a ser o foco principal desta abordagem, pois através dela são realizadas as duas principais tarefas do reconhecimento, a localização e a classificação dos elementos. Para que isso fosse possível, se fez necessária a obtenção de córpus previamente anotados, em nosso caso, córpus que possuem expressões temporais devidamente localizadas e classificadas. Em posse dos córpus, realizamos o treinamento dos dados utilizando diferentes tipos de marcações, bem como diferentes modelos de treinamento, como *chunking* e Mo-

delos de Markov. Para avaliação, utilizamos medidas de precisão, cobertura e medida F, a fim de obter o real desempenho do sistema.

3 Reconhecimento Automático de Expressões Temporais

Para a realização da tarefa de reconhecimento automático de expressões temporais, foram estudados diferentes formas de abordar o problema, tornando essenciais os estudos na área de aprendizado de máquina supervisionado. Esses estudos abrangem desde o formato adequado dos dados para realizar o treinamento, até os próprios algoritmos de classificação. Nas seções subsequentes são apresentados estes formatos, modelos e as medidas de avaliação utilizadas.

3.1 Formatos

Os dados usados para treinamento podem possuir várias informações. Essas informações são expressas através dos diferentes formatos em que os dados podem ser etiquetados. Neste trabalho, realizamos três tipos de etiquetagens, que nos permitiram configurar os dados de acordo com a necessidade. Tratamos também com a marcação original dos córpus usados como base dos treinamentos.

3.1.1 IOB

Os elementos presentes nos córpus de treinamento obtidos, possuem uma localização temporal definida a partir do início e fim de uma *tag* de marcação, como por exemplo em "Segundo colocado <ET> em 2004 </ET>, Sales processou seu adversário...". Porém, uma forma mais clara de definirmos a localização de uma expressão temporal, pode se dar através da utilização da marcação IOB. Esta marcação irá atribuir para cada elemento no texto uma de suas três *tags* I (*inside*), O (*outside*) e B (*begin*). Um elemento marcado com a *tag* B, irá representar o início de uma sequência selecionada, no nosso caso, o início de uma expressão temporal. Ao ser etiquetado com a *tag* I, o elemento é considerado parte da sequência, uma continuidade que teve início com a *tag* B. Por fim, todo elemento marcado com a *tag* O, será considerado fora da sequência de seleção. No nosso contexto, todo elemento que não se enquadra no conceito de expressão temporal. Um exemplo dessa utilização, aplicada ao contexto de expressão temporal, pode ser conferido na seguinte frase: "Acontecerá-O hoje-B o-O discurso-O do-O presidente-O,-O às-B 19-I horas-I,-O ele-O se-O pronunciará-O.-O".

3.1.2 Etiqueta Morfossintática

A Etiqueta Morfossintática é responsável pela classificação dos elementos de um texto através das classes gramaticais como: verbo, adjetivo, artigo, substantivo. Ela é usada em diversas tarefas da área de Linguagem Natural, como no processo de POS *tagging*, usualmente executado por classificadores que utilizam Modelos Ocultos de Markov (veja 3.3.1). Sua utilização pode ser empregada na seguinte frase: "Ele/PRO não/NEG queria/VB-D um/D-UM sítio/N fixo/ADJ". Esse tipo de etiqueta, permite uma melhor definição dos elementos no texto, especificando seus significados gramaticais dentro de seus contextos.

3.1.3 Etiqueta ET

Uma das etiquetas utilizadas como base de treinamento foi a própria classificação das expressões temporais encontradas nos córpus utilizados como base. Essa etiqueta permitiu a construção de um modelo anotador capaz de localizar e classificar as expressões temporais. Essa simultaneidade se deu ao fato de que ao classificar uma determinada expressão temporal, estamos, ao mesmo tempo, localizando-a no texto. A etiqueta é composta, na maioria das vezes, pelo tipo e subtipo das expressões temporais, porém, alguns tipos não apresentam subtipo, como é o caso de DURAÇÃO. As palavras que não são classificadas com expressão temporal, são anotadas com O, baseado no contexto apresentado nas etiquetas IOB. O modo como se apresenta estas etiquetas é apresentado no Exemplo abaixo.

```
Abadia/O esta/O envolvido/O com/O o/O trafico/O
desde/TEMPO_CALEND+DATA 1986/TEMPO_CALEND+DATA ./O
```

3.2 Córpus

Sendo o objetivo do presente trabalho a utilização de aprendizado de máquina supervisionado como meio para a construção de um reconhecedor de expressões temporais, fica clara a necessidade da utilização de dados pré-definidos. Esses dados estão presentes em córpus anotados, ou seja, córpus com a marcação correta dos dados contidos nele. Este trabalho teve duas fontes principais de córpus. A primeira delas é o Segundo HAREM, através do seu repositório na web[1]. No pacote disponibilizado no site existem dois córpus anotados, um principal possuindo 87549 palavras e outro menor com 15295 palavras. Ambos os córpus utilizam como marcação as *tags* <EM>, que referenciam entidades mencionadas. Por esse motivo, eles necessitam de um tratamento diferente para a extração das expressões temporais. No córpus principal as expressões temporais são localizadas e

[1] http://www.linguateca.pt/

classificadas, porém, utiliza-se para a classificação apenas o tipo e, quando existente, o subtipo da expressão, como no Exemplo abaixo.

```
<EM ID=‘‘dav-188222-189’’ CATEG=‘‘TEMPO’’ TIPO=‘‘TEMPO_CALEND’’
SUBTIPO=‘‘DATA’’>de 1995</EM>
```

O segundo córpus teve melhor especificação das expressões temporais, adicionando os atributos TEMPO_REF, SENTIDO e VAL_DELTA, como mostrados no Exemplo abaixo.

```
<EM ID=‘‘hub-21881-189’’ CATEG=‘‘TEMPO’’ TIPO=‘‘TEMPO_CALEND’’
SUBTIPO=‘‘DATA’’ TEMPO_REF=‘‘ENUNCIACAO’’ SENTIDO=‘‘POSTERIOR’’
VAL_DELTA=‘‘A1MOSODOHOMOSO’’>no proximo ano</EM>
```

Por se tratar da anotação de entidades mencionadas, ambos os córpus possuem diversas etiquetas das quais não utilizamos neste trabalho, como anotações de pessoa, local, valor monetário, por isso tivemos a necessidade de filtrar os córpus.

A segunda fonte foi obtida através do córpus disponibilizado pelo projeto Sucinto, na web[2]. O córpus possui uma marcação definida através da etiqueta <ET>, referenciando expressão temporal, isto é, um córpus específico na anotação das mesmas. Contendo um total de 55176 palavras, este córpus é focado no tratamento de expressões temporais, trazendo uma classificação bastante refinada, como pode ser visto no Exemplo abaixo:

```
<ET ID=‘‘07’’ TIPO=‘‘TEMPO_CALEND’’ SUBTIPO=‘‘DATA’’
TEMPO_REF=‘‘ENUNCIACAO’’
SENTIDO=‘‘POSTERIOR’’ VAL_DELTA=‘‘A0MOSOD1HOMOSO’’>de amanha</ET>.
```

3.3 Modelos

Para a realização dos treinamentos executados no processo de construção do reconhecedor temporal, foi necessário estudarmos os diferentes modelos possíveis. Entre eles estão os Modelos de Markov Ocultos (HMM, *Hidden Markov Models*) e o *Chunking* que serão abordados nas subseções a seguir.

3.3.1 Modelos de Markov Ocultos

O Modelo de Markov Oculto é uma ferramenta utilizada para realizar a descoberta de uma sequência de estados dentro de um modelo, tendo como parâmetros de análise os símbolos resultantes das transições feitas entre os estados e as suas probabilidades.

[2]http://www2.icmc.usp.br/ taspardo/sucinto/cstnews.html

Essa abordagem se difere do conceito de Cadeias de Markov, onde as transições são feitas sabendo para quais estados. Nas Cadeias de Markov temos ainda a propriedade Horizonte Limitado, onde, para a descoberta do próximo estado, usamos a informação do estado anterior. Em acréscimo a essa abordagem, existe a chamada Cadeia de Markov de ordem 2, onde utilizamos a informação dos dois estados anteriores para a definição do próximo. Esta aborgagem ocorre no processamento do *TrigramTagger* que será apresentado no Capítulo 4 como forma de resolução da tarefa proposta.

Um modelo HMM apresenta os seguintes elementos: um conjunto de estados (S), as probabilidades iniciais para cada estado (Π), um alfabeto de saída (K), um conjunto de probabilidades para a transição entre os estados (A) e um conjunto de probabilidades de um símbolo ser originado a partir de uma transição (B). Diante disso, existem três fundamentais questões a serem descobertas sobre um HMM.

A primeira questão é: A partir de um modelo μ = (A, B, Π), de que forma calculamos eficientemente a probabilidade de uma determinada observação?

A segunda questão é: Através de uma sequência de observações e um modelo μ, de que forma escolhemos a melhor sequência de estados para explicar as observações?

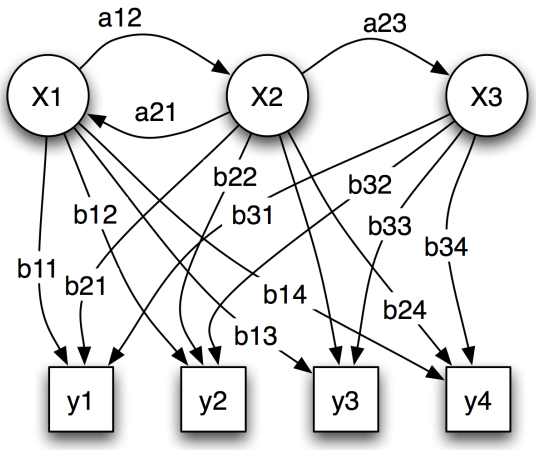
A terceira questão é: Dada uma sequência de observações e um espaço de modelos possíveis encontrados pela variação dos parâmetros μ = (A, B, Π), como encontramos o modelo que melhor explica os dados observados?

Dentre essas questões, a que vem a ser mais relevente a este trabalho é a segunda, pois se assemelha à forma como buscamos realizar a anotação das expressões temporais. Passando para o contexto de reconhecimento de expressões temporais, encontrar a melhor sequência de estados significa encontrar a melhor sequência de etiquetas para as palavras de um texto, como mostra a Figura 6. Nela podemos observar a sequência de estados, representados por x1, x2 e x3, juntamente com suas sequências de probabilidades de transição, representadas pelo a. Cada transição entre estados gera um símbolo, representados pelo y, cada estado possui uma probabilidade de gerar cada símbolo, representado pelo b. Para a resolução desta questão, utiliza-se, na maioria das vezes, o algoritmo de Viterbi, conhecido por possuir uma solução ótima recursiva para a descoberta da melhor sequência oculta dada uma sequência de observações (MARTIN; JURAFSKY, 2000).

3.3.2 Chunking

Chunking é conhecida como uma tarefa de pré-processamento de um parser, que tem por objetivo a divisão de um texto em frases, de forma que as palavras relacionadas sintaticamente façam parte de um mesmo conjunto (SANG; BUCHHOLZ, 2000). Sua utilização se dá de modo sequencial, analisando os elementos através de suas etiquetas e assim agrupando-os. Normalmente presente em tarefas de reconhecimento de sintagmas

Figura 6 – Modelos Ocultos de Markov(Wikipedia).



nominais.

3.4 Medidas de Avaliação

Para obter o desempenho do anotador criado, se faz necessária a condução de experimentos e avaliações dos resultados alcançados. Perante isso, escolhemos algumas medidas de avaliação buscando abranger da melhor forma todos os aspectos do sistema. Visando descobrir a taxa de acertos obtida através da etiquetagem, ou seja, quantos elementos foram corretamente anotados, decidiu-se pela utilização da medida de precisão. Por outro lado, é necessário saber se essa etiquetagem está abrangendo a expressão temporal por completo, ao invés de partes dela. Para isso foi escolhida a medida de cobertura. Por fim, para obter uma medida que englobe ambos resultados, gerando assim uma visão geral do desempenho do sistema, foi escolhida a medida F, que utiliza em seu cálculo as duas medidas anteriores.

Para realizar o cálculo das três medidas, é necessário primeiramente analisar os resultados obtidos com o anotador em comparação aos dados corretos. A partir desta análise, calculamos o número de verdadeiros positivos (vp), verdadeiros negativos (vn), falsos positivos (fp) e falsos negativos (fn). Verdadeiros positivos são considerados todas as marcações corretas, ou seja, as quais foram classificadas pelo anotador e se encontram da mesma maneira na referência. Verdadeiros negativos são todas as marcações não classificadas pelo anotador que também não estão classificadas na referência. Falsos positivos são os dados classificados pelo anotador que não possuem classificação na referência. Por fim, os falsos negativos são os dados não classificados pelo anotador, porém classificados pela referência.

Tendo calculado os valores acima, passamos para o cálculo das medidas, inici-

ando pela medida de precisão, ela faz a divisão dos verdadeiros positivos pela soma dos verdadeiros positivos e falsos positivos, como mostra o exemplo abaixo.

$$Precisão = \frac{vp}{vp + fp}$$

O cálculo da medida de cobertura realiza a divisão dos verdadeiros positivos pela soma dos verdadeiros positivos e dos falsos negativos, como reproduz o exemplo abaixo.

$$Cobertura = \frac{vp}{vp + fn}$$

Finalmente, o cálculo da medida f é realizado tendo como base as duas medidas anteriores, sendo assim, através da utilização de uma variável como peso, é realizada a medição apresentada no exemplo a seguir. A variável de peso ao receber valores menores que 1, dá ênfase à precisão e ao receber valores maiores que 1, enfatiza a cobertura, sendo o valor 1 o peso de igualdade.

$$MedidaF_{\beta} = (1 + \beta^2) \times \frac{Precisão \times Cobertura}{\beta^2 \times Precisão + Cobertura}$$

Outra forma de avaliação normalmente utilizada em tarefas relacionadas à etiquetagem, é a acurácia. Ela faz a divisão de todos os acertos obtidos pelo anotador pela soma dos acertos e erros. O cálculo está representado no exemplo abaixo. Pra a avaliação dos modelos criados neste trabalho, não utilizamos a medida de acurácia, isso porque a maioria dos elementos anotados na tarefa não são expressões temporais, sendo a porcentagem de elemento muito baixa. Com isso, o valor da acurácia não remete diretamente ao desempenho do sistema.

$$Acurácia = \frac{tp + tn}{tp + tn + fp + fn}$$

4 Experimentos e Resultados

Tendo como base a metodologia apresentada e os conceitos relacionados à área de Linguagem Natural, foram realizados diversos experimentos com a finalidade de alcançarmos os objetivos desta monografia. Para que pudéssemos obter maior flexibilidade no tratamento de texto, sendo este o principal foco do trabalho, optamos pela utilização da linguagem de programação Python em sua versão 2.7.3.

> *Python is an interpreted, interactive, object-oriented programming language. It provides high-level data structures such as list and associative arrays (called dictionaries), dynamic typing and dynamic binding, modules, classes, exceptions, automatic memory management, etc.*(SANNER et al., 1999)

No contexto de linguagem natural, utilizamos o conjunto de ferramentas open source, NLTK 2.0, desenvolvido na linguagem Python. O NLTK é composto por mais de 50 córpus e diversas bibliotecas, permitindo um alto nível em processamento de texto, como em tarefas de classificação e anotação. Os experimentos tiveram início a partir da construção de uma baseline, para então buscarmos abordagens mais complexas, que fizessem uso de aprendizado de máquina supervisionado através dos conjuntos de córpus anotados.

4.1 Baseline

Para dar início ao trabalho de implementação, foi planejada a construção de uma *baseline*, ou seja, uma aplicação trivial que utilizamos como base para comparação de desempenho. Neste experimento fizemos uso dos córpus anotados, onde retiramos as informações sobre a frequência em que as palavras apareciam marcadas como expressão temporal. Desta forma, se determinada palavra possuísse uma frequência maior que 50%, ela seria considerada uma expressão temporal. Essa aplicação teve como objetivo apenas a localização das expressões temporais, não obtendo resultados significativos. A codificação foi dividida em duas partes, onde a primeira tinha foco na descoberta das expressões temporais presentes nos córpus anotados e a segunda realiza o cálculo das frequências. Para facilitar a obtenção da distribuição de frequência, utilizamos o módulo de probabilidade *FreqDist*, presente no NLTK que calcula os valores através das entradas passadas em formatos de tupla, contendo a palavra e sua definição. Como resultado,

obtivemos um localizador temporal que recebe como entrada um texto como no Exemplo 8 e retorna o texto anotado, como mostra o Exemplo 9.

Exemplo 8 – Exemplo.

```
``Hoje, dia 19 de dezembro de 2001, acontecera a grande maratona que
envolve diversos corredores de diferentes paises, entre eles esta
o Brasil participando pela quarta vez. Este ano o Brasil ja ganhou
tres medalhas, em maratonas ja sao duas, vamos esperar entao bons
resultados da amarelinha hoje.''
```

Exemplo 9 – Retorno do Baseline.

```
``<ET> Hoje dia 19 </ET> de <ET> dezembro </ET> de <ET> 2001 </ET>
acontecera a grande maratona que envolve diversos corredores
de diferentes paises entre eles esta o Brasil participando pela
quarta vez Este <ET> ano </ET> o Brasil ja ganhou tres medalhas
em maratonas ja sao duas vamos esperar entao bons resultados
da amarelinha <ET> hoje </ET>''
```

Como o algoritmo baseia-se apenas no cálculo da frequência, palavras-chave como "hoje", "ano" e "dezembro" são facilmente localizadas, porém, palavras como "de" e "este", mesmo aplicadas no contexto de uma expressão temporal, acabam não sendo localizadas. Para a anotação da frase recebida como entrada, foram retiradas as pontuações, a fim de facilitar a localização. Com este *baseline* obtivemos os resultados apresentados na Tabela 4.

Tabela 4 – Resultados do Baseline.

	Precisão	**Cobertura**	**Medida F**
Baseline	0.45	0.47	0.46

4.2 IOBTagger

Com a construção do IOBTagger visamos melhorar a localização das expressões temporais. Para que isso fosse possível, modificamos o formato de marcação dos córpus. Dessa forma, o texto, que antes era composto pelas *tags* <ET> ou <EM>, passa a conter apenas B, I ou O, como mostra a transição do Exemplo 10 para o Exemplo 12.

Esse novo formato nos permite realizar o treinamento sobre os elementos no texto. Dividimos o conteúdo gerado em dois conjuntos, 80% das sentenças para treinamento e

Exemplo 10 – Formato original dos Córpus.

```
<ET ID=‘‘02’’ TIPO=‘‘TEMPO_CALEND’’ SUBTIPO=‘‘DATA’’>No mesmo dia</ET>
em que Lula se comprometeu a nao atacar, o adversario tucano Geraldo
Alckmin elevou o tom do seu discurso. Alckmin disse <ET ID=‘‘03’’
TIPO=‘‘TEMPO_CALEND’’ SUBTIPO=‘‘DATA’’>neste domingo</ET> que Lula
deu as costas para o povo brasileiro, para a Justica e para os bons
costumes.
```

Exemplo 11 – Formato IOB dos Córpus.

```
No/B mesmo/I dia/I em/O que/O Lula/O se/O comprometeu/O a/O nao/O
atacar/O,/O o/O adversario/O tucano/O Geraldo/O Alckmin/O elevou/O
o/O tom/O do/O seu/O discurso/O./O Alckmin/O disse/O neste/B domingo/I
que/O Lula/O deu/O as/O costas/O para/O o/O povo/O brasileiro/O,/O
para/O a/O Justica/O e/O para/O os/O bons/O costumes/O./O
```

20% para teste. Fazendo uso dos *taggers*(anotadores) disponibilizados pelo NLTK, treinamos os dados. Inicialmente, utilizamos o *DefaultTagger*, onde definimos a *tag* mais frequente. Para a aplicação em questão utilizamos a *tag* O. Após esse primeiro treinamento, passamos para o *UnigramTagger*, o qual recebe o conjunto de treinamento e conta com o *DefaultTagger* como complemento. São passadas ao *DefaultTagger* as palavras das quais o *UnigramTagger* não consegue atribuir uma etiqueta. O mesmo processo acontece no treinamento do *BigramTagger*, que além do conjunto de treinamento, tem como complemento o *UnigramTagger*. E por fim, é feito o treinamento do *TrigramTagger* com o conjunto de treinamento e o *BigramTagger* como complemento. Os resultados demonstraram um declínio na medida f, porém, um aumento na precisão. A Tabela 5 mostra a avaliação do experimento através das medidas de Precisão, Cobertura e Medida-F.

Tabela 5 – Resultados do IOBTagger.

	Precisão	**Cobertura**	**Medida F**
IOBTagger	0.66	0.11	0.20

4.3 POS+IOBTagger

Buscamos uma abordagem diferente para a resolução do problema. Optamos por adicionar uma segunda etiqueta às palavras. Decidimos pelo uso de uma etiqueta morfossintática e para isso, as palavras presentes nos córpus deveriam passar por um anotador morfossintático, tarefa conhecida como *part-of-speech tagging* (POS *tagging*). Essa tarefa realiza a atribuição de uma classe gramatical a cada palavra contida no texto, resultando

as etiquetas morfossintáticas necessárias ao treinamento. Para que pudéssemos obter as novas etiquetas, foi necessário o treinamento de um anotador morfossintático. Para isso, utilizamos como base os córpus anotados em português presentes no NLTK. Os córpus selecionados foram o Floresta e o Mac Morpho, ambos com anotação própria, e por esse motivo, passados para treinamento separadamente. Foram realizados dois tipos de treinamentos, primeiramente utilizamos os *taggers* disponibilizados pelo NLTK e posteriormente foi utilizado o *tagger* VLMMTagger (KEPLER; FINGER, 2010). Realizado o primeiro treinamento do anotador morfossintático utilizando os *taggers* NLTK, foi feita a anotação das palavras presentes nos córpus de anotação temporal, o que resultou nas POS *tags*. Essas etiquetas foram concatenadas às etiquetas IOB, sendo separadas pelo símbolo "=" e deixando as palavras como no formato da seguinte frase:

Exemplo 12 – Formato POS+IOB.

```
‘‘...em/B=P Novembro/I=NPR de/I=P 1534/I=NUM ,/O=, colocou/O=VB-D
Henrique/O=NPR e/O=CONJ os/O=D-P seus/O=PRO\$-P sucessores/O=N-P
a/O=P+D-F lideranca/O=N...’’
```

Possuindo esse novo córpus anotado, novamente o dividimos em 80% para treinamento e 20% para teste e então treinamos o anotador temporal da mesma forma descrita anteriormente. Os resultados apresentados na Tabela 6 mostram os desempenhos utilizando os diferentes córpus anotados, Floresta e Mac Morpho, como treinamento do anotador morfossintático.

Tabela 6 – Resultados do POSTagger com diferentes córpus anotados.

	Precisão	**Cobertura**	**Medida F**
Experimento 2 (Floresta)	0.70	0.38	0.49
Experimento 2 (Mac Morpho)	0.69	0.36	0.48

Como conclusão da utilização do VLMMTagger, obtivemos uma melhora nos resultados, que são apresentados na Tabela 7:

Tabela 7 – Resultados do IOB+POSTagger usando VLMMTagger

	Precisão	**Cobertura**	**Medida F**
IOB+POSTagger	0.79	0.56	0.65

Constatando uma evidente melhora nos resultados obtidos através do VLMMTagger, decidimos realizar ambos treinamentos fazendo uso do mesmo. Desta forma, o treinamento do anotador morfossintático e do anotador temporal foram construídos e então avaliados da mesma maneira anterior, obtendo os resultados mostrados na Tabela 8:

Tabela 8 – Resultados do IOB+POSTagger Final.

	Precisão	**Cobertura**	**Medida F**
IOB+POSTagger	0.77	0.81	0.79

4.4 ETTagger

Após obter resultados mais aproximados ao desejável quanto a localização temporal e termos definido o VLMMTagger como sendo a melhor escolha entre as opções utilizadas, decidimos realizar o treinamento usando como etiqueta a classificação das expressões temporais. Sendo assim, foi construído um reconhecedor de expressões temporais, pois torna-se responsável não só pela localização como pela classificação das mesmas.

Para a realização do experimento foi necessário transformar os córpus, deixando apenas as palavras com suas respectivas classificações. Existem diferenças entre as classificações dos córpus, porém, a classificação de tipo e subtipo ocorrem da mesma maneira em todos. Dessa forma, as palavras identificadas como ET estavam agora etiquetadas com a classificação de seu tipo e subtipo, sendo essas categorias ligadas pelo símbolo "+", como em "10/TEMPO_CALEND+DATA de/TEMPO_CALEND+DATA Agosto/TEMPO_CALEND+DATA". E o restante, não classificadas como ET, etiquetadas com o O.

O treinamento foi realizado da mesma maneira que nos experimentos anteriores, passando 80% das sentenças anotadas ao *tagger* e reservando 20% para a realização do teste. Como resultados, obtivemos as medições apresentadas na Tabela 9. Essa medida é resultado da média feita em cima de dez testes realizados alternando aleatoriamente os 80% e 20% retirados do córpus. Obsevamos uma diminuição na Medida F, era esperada, uma vez que o anotador tinha de acertar tanto a localização quanto a classificação.

Tabela 9 – Resultados do ETTagger.

Execução	**Precisão**	**Cobertura**	**Medida F**
1	0.65	0.68	0.67
2	0.65	0.73	0.69
3	0.72	0.72	0.72
4	0.70	0.75	0.73
5	0.70	0.74	0.72
6	0.69	0.75	0.72
7	0.73	0.72	0.72
8	0.69	0.75	0.71
9	0.68	0.73	0.71
10	0.70	0.71	0.70
Média	0.69	0.73	0.71

4.5 POS+ETTagger

Como realizado no modelo POS+IOBTagger, decidimos adicionar a POS *tag* à etiqueta ET, com o intuito de obtermos melhores resultados aos anteriores apresentados, dessa maneira criamos um novo córpus de treinamento utilizando as duas *tags*, ET e POS. Para realizar os testes foram passadas apenas as etiquetas de classificação, retirando as POS *tags*, uma vez que as mesmas não são o foco dos acertos, mas sim as classificações corretas. O modelo foi executado 10 vezes e os resultados estão presentes na Tabela 10, apresentando ligeira melhora em relação aos testes anterores, constatando visível melhora com a utilização da *tag* morfossintática.

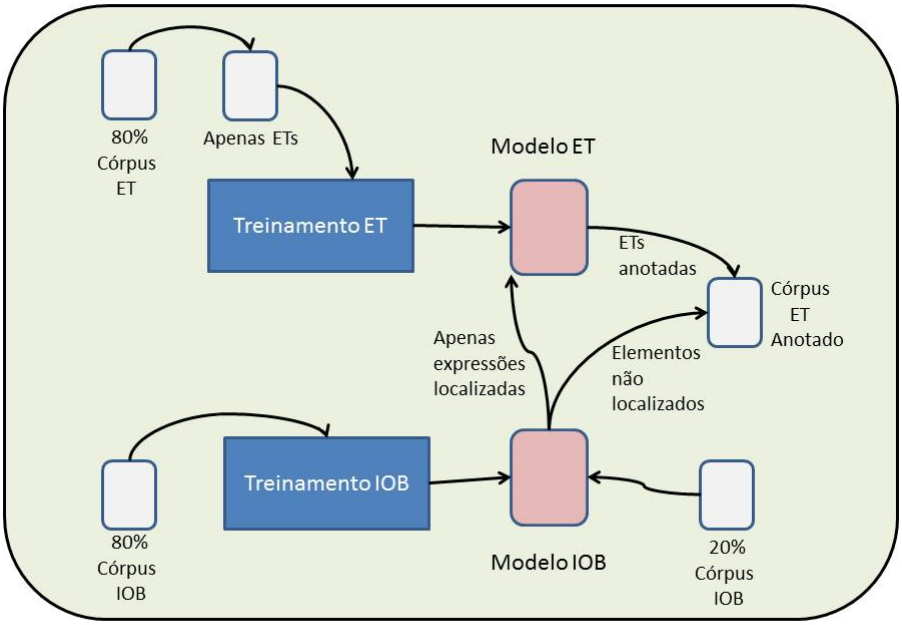
Tabela 10 – Resultados do POS+ETTagger.

Execução	**Precisão**	**Cobertura**	**Medida F**
1	0.72	0.80	0.76
2	0.71	0.75	0.73
3	0.72	0.74	0.73
4	0.68	0.79	0.73
5	0.69	0.77	0.73
6	0.73	0.76	0.74
7	0.74	0.78	0.76
8	0.71	0.80	0.75
9	0.73	0.77	0.75
10	0.70	0.76	0.73
Média	0.71	0.77	0.74

4.6 IOB+ETTagger

Como última tentativa de obter melhora nos resultados até então apresentados, foi elaborado o IOB+POSTagger, que busca a união dos resultados alcançados na localização com os apresentados pelo reconhecedor. O experimento foi organizado da seguinte maneira. Primeiramente foram selecionados os córpus construídos anteriormente. O que possui a anotação IOB+POS e o que possui apenas a classificação. A partir destes córpus, foram selecionados 80% das sentenças presentes em cada um, formando seus conjuntos de treinamento e os outros 20% destinados para teste. Dos 80% selecionados do córpus de classificação, foi feita uma filtragem, construindo um novo córpus constituído apenas pelas sentenças anotadas como expressão temporal. Dos 20% selecionados do córpus IOB+POS, foram retiradas apenas as palavras e passadas ao VLMMTagger, já previamente treinado, resultando num novo córpus anotado. O córpus anotado apenas com expressões temporais foi utilizado como base para o treinamento de um novo anotador através do *TrigramTagger*. Utilizando o córpus originado da anotação feita pelo VLMMTagger, passamos para um processo dividido em duas partes, que por fim construíram um novo córpus anotado. A primeira parte do processo é a adição dos elementos etiquetados

Figura 7 – Processo IOB+ETTagger



com a *tag* O a um novo córpus. A segunda parte acontece quando durante o percurso é encontrada uma expressão temporal, ou seja, uma palavra etiquetada com a *tag* 'B' ou 'I'. Ela é então passada a uma lista que irá guardar toda a sentença temporal. Ao acabar a sentença, as palavras contidas nela são passadas ao modelo treinado pelo *TrigramTagger*. O resultado é adicionado ao novo córpus. O processo deste modelo pode ser visto na Figura 7.

No final do procedimento, temos um córpus com as palavras temporais classificadas e o restante das palavras etiquetadas com O. Utilizando os 20% do córpus de classificação separado no início do procedimento, foi realizada a avaliação sobre o córpus gerado no procedimento. Os resultados das anotações feitas durante o processo estão presentes na Tabela 11. A comparação de todos os experimentos pode ser visualizada na Tabela 12.

Tabela 11 – Resultados Experimento Final.

	Precisão	**Cobertura**	**Medida F**
Treinamento VLMMTagger	0.77	0.81	0.79
IOB+ETTagger	0.65	0.79	0.71

Tabela 12 – Resultados dos Experimentos.

	Modelo	**Precisão**	**Cobertura**	**Medida F**
Localização	Baseline	0.45	0.47	0.46
	IOBTagger	0.66	0.11	0.20
	POS+IOBTagger	0.77	0.81	0.79
Reconhecimento	ETTagger	0.69	0.73	0.71
	POS+ETTagger	0.71	0.77	0.74
	IOB+ETTagger	0.65	0.79	0.71

5 Considerações Finais

Nesta monografia apresentamos a proposta de resolução da tarefa de Reconhecimento Automático de Expressões Temporais. Para isso foi feita uma revisão das áreas de Reconhecimento de Entidades Mencionadas e Reconhecimento de Expressões Temporais. Abordamos o conceito de Expressão Temporal, seus tipos e ambiguidades, bem como mostramos a importância do seu reconhecimento em diferentes tarefas de Extração de Informação. Foram também revisadas as diferentes abordagens existentes para a resolução da tarefa, apresentando aplicações que utilizam métodos simbólicos e outras aprendizado de máquina.

Para a resolução da tarefa propomos o treinamento de modelos computacionas através da técnica de aprendizado de máquina supervisionado. Seu treinamento foi realizado a partir de córpus anotados para a língua portuguesa adquiridos em repositórios online. Os treinamentos envolveram diferentes tipos de marcações como IOB e Etiqueta Morfossintática, gerando diferentes modelos. Como resultados, obtivemos modelos que reconhecem as expressões temporais corretamente, partindo das classificações definidas. Nosso melhor modelo alcançou 74% de medida F, o que, por não fazer uso de nenhjum pré-conhecimento especialista, se mostra um ótimo resultado frente às abordagens atuais. Além disso, este trabalho nos proporcionou a participação no WPLN 2012 - *Workshop on Natural Language Processing*, realizado em Montevidéu.

Uma das maiores dificuldades encontradas no contexto de aprendizado de máquina supervisionado, é a necessidade de uma considerável quantia de dados anotados para realizar o treinamento. Com isso, como trabalhos futuros, pode-se buscar mais fontes de dados a fim de refinar os modelos construídos e alcançar melhores resultados.

Referências

AHN, D.; RANTWIJK, J.; RIJKE, M. A cascaded machine learning approach to interpreting temporal expressions. Citeseer, 2007. Citado 3 vezes nas páginas 11, 35 e 36.

AMANCIO, M. A. *Elaboração Textual via Definição de Entidades Mencionadas e de Perguntas aos Verbos em Textos Simplificados do Português*. Dissertação (Mestrado) — Instituto de Ciências Matemáticas e de Computação, 2009. Citado na página 22.

BAPTISTA, C. H. e. N. M. J. Identificação, classificação e normalização de expressães temporais do português: A experiência do segundo harem e o futuro. In: _____. [S.l.]: Cristina Mota and Diana Santos, 2008. cap. 2, p. 33–54. Citado na página 23.

CARVALHO, P. et al. Segundo harem: Modelo geral, novidades e avaliação. In: _____. [S.l.]: Cristina Mota e Diana Santos, 2008. cap. 1, p. 11–31. Citado 4 vezes nas páginas 11, 22, 27 e 30.

CARVALHO, W. S. *Reconhecimento de entidades mencionadas em português utilizando aprendizado de máquina*. Dissertação (Mestrado) — Instituto de Matemática e Estatística da Universidade de São Paulo, 2012. Citado na página 31.

CHARNIAK, E. A maximum-entropy-inspired parser. In: MORGAN KAUFMANN PUBLISHERS INC. *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*. [S.l.], 2000. p. 132–139. Citado na página 35.

CHOWDHURY, G. Natural language processing. *Annual review of information science and technology*, Wiley Online Library, v. 37, n. 1, p. 51–89, 2003. Citado na página 19.

CRAVEIRO, O.; MACEDO, J.; MADEIRA, H. Portexto: sistema de anotação/extracção de expressões temporais. In: _____. [S.l.]: Cristina Mota and Diana Santos, 2008. cap. 8, p. 159–170. Citado na página 32.

FILHO, H. C. *CoppeTER–Reconhecimento e Normalização de Expressões Temporais em Português*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2011. Citado 4 vezes nas páginas 11, 19, 33 e 35.

HAGÉGE, C.; BAPTISTA, J.; MAMEDE, N. Proposta de anotação e normalização de expressões temporais da categoria tempo para o segundo harem. In: _____. [S.l.]: Santos, 2008. cap. Apêndice B, p. 289–308. Citado na página 28.

KEPLER, F. N.; FINGER, M. Variable-length markov models and ambiguous words in portuguese. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas*. [S.l.], 2010. p. 15–23. Citado na página 48.

LOUREIRO, J. *Reconhecimento de Entidades Mencionadas (Obra, Valor, Relações de Parentesco e Tempo) e Normalizaçao de Expressoes Temporais*. Tese (Doutorado)

— Masters thesis, Technical University of Lisbon, Instituto Superior Técnico, Lisboa, Portugal, 2007. Citado na página 21.

MARTIN, J. H.; JURAFSKY, D. *Speech and language processing*. [S.l.]: prentice hall, 2000. Citado na página 42.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. In: _____. [S.l.]: Oliveira, 2003. cap. 4, p. 89–107. Citado na página 31.

MOTA, C. et al. É tempo de avaliar o tempo. In: _____. [S.l.: s.n.], 2008. cap. 3, p. 55–75. Citado na página 22.

NOBATA, C.; SEKINE, S. Automatic acquisition of patterns for information extraction. *Proceeding of the ICCPOL1999*, 1998. Citado na página 21.

OSORIO, F.; VIEIRA, R. Sistemas híbridos inteligentes. In: *ENIA-Encontro Nacional de Inteligência Artificial/Congresso da SBC*. [S.l.: s.n.], 1999. p. 1–59. Citado na página 30.

REPENTINO. *REPENTINO. Disponível em:*. 2005. http://labclup.letras.up.pt/repentino/. Citado na página 22.

SANG, E. F. T. K.; BUCHHOLZ, S. Introduction to the conll-2000 shared task: Chunking. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*. [S.l.], 2000. p. 127–132. Citado na página 42.

SANNER, M. F. et al. Python: a programming language for software integration and development. *J Mol Graph Model*, Citeseer, v. 17, n. 1, p. 57–61, 1999. Citado na página 45.

SOUTO, M. de et al. Técnicas de aprendizado de máquina para problemas de biologia molecular. *III Jornada de Inteligência Artificial*, 2003. Citado na página 31.

TIMEX2. *Timex2. Disponível em:*. http://www.timexportal.info/timex2. Citado na página 26.

WOLF, M.; WICKSTEED, C. Date and time formats. *W3C NOTE NOTE-datetime-19980827, August*, 1998. Citado na página 26.