

# Identificação de Orações com ETL

Bernardo A. Pires

Eraldo R. Fernandes

Projeto apresentado na disciplina de Aprendizado Máquina II ministrada pelo Prof. Ruy L. Milidiú na Pós-Graduação do Departamento de Informática da Pontifícia Universidade Católica do Rio de Janeiro.

Julho de 2009

## 1 Introdução

A tarefa de *identificação de orações* (*clause identification*) consiste em determinar as orações de uma frase. Identificar orações em uma frase é um tipo de *análise rasa* (*shallow parsing*), assim como *text chunking* [San00]. Este tipo de informação é muito importante na resolução de tarefas mais complexas como identificação de papéis semânticos (*semantic role labeling*) [MCLS08] e construção de árvore de dependência [Niv05].

Vários sistemas baseados em *aprendizado de máquina* foram propostos para este problema. A tradicional competição da *Conference on Computational Natural Language Learning* (CoNLL) em 2001 enfocou esta tarefa [SD01]. Nesta competição foi disponibilizado um corpus com aproximadamente 200 mil palavras em língua inglesa. Este corpus foi gerado a partir de uma parte do *Penn Treebank* [MSM93], contendo textos do *Wall Street Journal*. O sistema [CMC05] que detém o melhor resultado para o corpus

CoNLL’2001, até o presente momento, é baseado em *Boosted Perceptrons*. Dois sistemas [CM01, CMPR02] baseados em *Boosted Trees* detêm, respectivamente, o segundo e o terceiro melhores resultados na identificação de orações para a língua inglesa.

Neste trabalho, apresentamos um extrator de orações e o avaliamos no corpus CoNLL’2001. O sistema proposto é baseado no algoritmo *Entropy Guided Transformation Learning* (ETL) [dSM09]. ETL é um novo algoritmo de aprendizado de máquina que generaliza o algoritmo *Transformation Based Learning* (TBL) [Bri95], resolvendo seu principal gargalo: a geração automática de gabaritos de regras. O ETL foi aplicado com sucesso para diversos problemas de linguística computacional [MdSD08a, MdSD08b, dSMR08, dSM09].

Os principais resultados alcançados com o sistema proposto são apresentados na Tabela 1, juntamente com os resultados do melhor sistema da CoNLL’2001 [CM01] e do sistema que detém o resultado no estado da arte [CMC05]. O desempenho do sistema proposto é competitivo com o estado da arte. Além disso, a modelagem ETL adotada é bastante simples e geral, podendo ser facilmente estendida para outros problemas de linguística computacional. As modelagens utilizadas nos sistemas com desempenho superior são bem complexas e utilizam características específicas do problema de identificação de orações.

<i>Sistema</i>	<i>Precisão</i>	<i>Abrangência</i>	$F_{\beta=1}$
Estado da arte	88,17	81,10	85,03
CONLL’2001	84,82	78,85	81,73
ETL	86,37	75,45	80,55

Tabela 1: Desempenho no corpus CoNLL’2001.

O restante deste documento está organizado da seguinte maneira. Na Seção 2 a tarefa de identificação de orações e o corpus utilizado são descritos. As técnicas de modelagem desenvolvidas neste trabalho são apresentadas na Seção 3. Uma modelagem alternativa, baseada em otimização combinatoria, é discutida na Seção 4. Na Seção 5 são descritos os experimentos computacionais realizados e os resultados obtidos. Finalmente, na Seção 6, são apresentadas as conclusões do trabalho e alguns trabalhos futuros promissores.

## 2 Tarefa e Corpus

A tarefa de identificação de orações consiste em determinar todas as orações de uma dada frase. Uma oração é uma sub-sequência de tokens de uma frase e é composta por um sujeito e um predicado. Uma mesma frase pode conter várias orações. Além disso, uma oração pode conter orações aninhadas. Veja na Figura 1 um exemplo de uma frase dividida em orações. Neste exemplo, parênteses são utilizados para identificar as orações, ou seja, eles não fazem parte das frases.

```
( Not everyone believes
  ( that
    ( the good times are over for shippers )
  )
  .
)
```

Figura 1: Frase do corpus CoNLL'2001 dividida em orações.

Neste trabalho o corpus CoNLL'2001 é utilizado para avaliar o desempenho do sistema proposto. Este corpus foi disponibilizado publicamente para a tradicional competição da *Conference on Computational Natural Language Learning* em 2001 [SD01]. Ele foi gerado a partir do *Penn Treebank* [MSM93] e contém textos em língua inglesa do *Wall Street Journal*. Na CoNLL'2001 a tarefa de identificação de orações é dividida em três sub-tarefas:

- Identificação de tokens que *iniciam uma oração*.
- Identificação de tokens que *terminam uma oração*.
- Identificação das orações completas.

O corpus disponibilizado contém anotações de *part-of-speech* (POS) e *text chunking*, além das anotações para as três sub-tarefas a serem resolvidas. O formato do corpus é ilustrado na Tabela 2, utilizando a frase do exemplo apresentado anteriormente. Como pode ser observado no exemplo, todas as informações do corpus são codificadas como uma classificação *token-a-token*. A coluna *Start* contém as classes *S* e *X* que identificam, respectivamente, os tokens que iniciam uma oração e os que não iniciam. A coluna *End* contém as classes *E* e *X* que identificam, respectivamente, os tokens que terminam uma oração e os que não terminam. A coluna *Clause* contém as classes *\**, (*S\**,

*\*S*), e qualquer combinação destas três classes. Esta última coluna codifica o conjunto completo de orações através deste esquema de parentização.

<i>Word</i>	<i>POS</i>	<i>Chunk</i>	<i>Start</i>	<i>End</i>	<i>Clause</i>
Not	RB	B-NP	S	X	(S*
everyone	NN	I-NP	X	X	*
believes	VBZ	B-VP	X	X	*
that	IN	B-SBAR	S	X	(S*
the	DT	B-NP	S	X	(S*
good	JJ	I-NP	X	X	*
times	NNS	I-NP	X	X	*
are	VBP	B-VP	X	X	*
over	IN	B-PP	X	X	*
for	IN	B-PP	X	X	*
shippers	NNS	B-NP	X	E	*S)S)
.	.	O	X	E	*S)

Tabela 2: Formato do corpus CoNLL’2001.

O corpus CoNLL’2001 é dividido em três partes: treino, desenvolvimento e teste. Informações sobre a dimensão e a composição destas partes são apresentadas na Tabela 3.

	<i>Sentenças</i>	<i>Tokens</i>	<i>S tokens</i>	<i>E tokens</i>	<i>Orações</i>
<i>Treino</i>	8.936	211.727	23.157	17.187	24.841
<i>Desenvolvimento</i>	2.012	47.377	5.089	3.737	5.418
<i>Teste</i>	1.671	40.039	4.497	3.364	4.856

Tabela 3: Dimensão e composição das três partes do corpus CoNLL’2001.

### 3 Modelagem

Devido à dificuldade da tarefa de identificação de orações, em geral, ela é dividida em sub-tarefas. Neste trabalho, a exemplo da estratégia utilizada na CoNLL’2001, esta tarefa é dividida em três sub-tarefas: (i) identificação de tokens que iniciam uma oração (*tokens S*); (ii) identificação de tokens que terminam uma oração (*tokens E*); e (iii) identificação das orações completas. Estas sub-tarefas são resolvidas sequencialmente e as informações produzidas nas primeiras sub-tarefas são utilizadas nas sub-tarefas posteriores. As

modelagens propostas neste trabalho, para cada uma destas sub-tarefas, são apresentadas a seguir.

### 3.1 Classificação de tokens $S$ e tokens $E$

As duas primeiras sub-tarefas (identificação de tokens  $S$  e tokens  $E$ ) são tratadas como problemas de classificação de tokens. Estas duas tarefas são modeladas de maneira direta utilizando-se o algoritmo ETL e o corpus CoNLL'2001. Para a primeira sub-tarefa, um modelo ETL é treinado para classificar cada token como  $S$  ou  $X$ . Analogamente, para a segunda sub-tarefa, um modelo ETL é treinado para classificar cada token como  $E$  ou  $X$ .

### 3.2 Identificação de orações completas

A última sub-tarefa é a mais importante e consiste em identificar as orações completas. Nesta fase, o sistema concilia os tokens  $S$  e  $E$ , obtidos nas fases anteriores, de modo a formar orações completas. São propostas diferentes modelagens para esta sub-tarefa. As modelagens propostas são denominadas: *ETL-pTk*, *ETL-Pair* e *ETL-Struc*. Estas modelagens são discutidas detalhadamente a seguir.

#### ***ETL-pTk*: classificador por token**

Nesta modelagem, ao contrário das outras abordagens propostas, a terceira sub-tarefa é tratada como um problema de classificação de tokens. Para isto, um modelo ETL é treinado para classificar cada token como:  $*$ ,  $(S^*$ ,  $*S)$ ,  $(S(S^*$ ,  $*S)S)$ , e qualquer outra combinação encontrada no corpus de treino.

Este classificador não identifica as orações diretamente. Ele realiza uma parentização da frase através de uma classificação por *token*. Desta forma, o sistema pode produzir uma classificação não consistente. Por isso, ao final da classificação, uma heurística simples é utilizada para resolver possíveis inconsistências.

#### ***ETL-Pair*: classificador de pares**

O modelo anterior utiliza uma abordagem simples e, conseqüentemente, não captura aspectos relevantes do problema. O problema de identificar orações não é, naturalmente, um problema de classificação de tokens. Por isso, um modelo mais poderoso é desenvolvido neste trabalho.

Neste modelo, um novo conjunto de treino é gerado após as etapas de identificação de tokens  $S$  e  $E$ . Para cada par de tokens classificados como  $S$  e  $E$ , um exemplo é gerado no novo corpus de treino. Cada novo exemplo é constituído pela união das informações (Word, POS e Chunk) dos dois tokens. A partir deste novo corpus, um classificador ETL binário é treinado para identificar se um par de tokens delimita ou não uma oração.

Este modelo é mais poderoso do que o anterior, pois sempre considera as informações do início e do final da oração candidata.

### ***ETL-Struc*: classificador de pares estruturado**

O algoritmo ETL possui uma característica muito poderosa que permite considerar a interdependência entre exemplos, de uma maneira muito simples. Para isto, basta agrupar os exemplos dependentes sequencialmente. Desta forma, o ETL considera, ao classificar um determinado exemplo, as informações dos exemplos vizinhos.

Para explorar esta característica, construímos uma variante do modelo anterior. Neste novo modelo, os exemplos (pares de tokens  $S$  e  $E$ ) oriundos de uma mesma frase são agrupados. Desta forma, o ETL pode explorar a dependência existente entre os exemplos da mesma frase.

### **3.3 Informações adicionais**

Além das informações fornecidas como entrada no corpus CoNLL'2001, algumas *informações adicionais* são geradas. Estas informações são geradas de maneira similar à proposta em [CMPR02]. Entretanto, no presente trabalho, apenas uma pequena parte das informações proposta por estes autores é utilizada.

As informações adicionais informam sobre a ocorrência de *elementos relevantes* em um determinado trecho da frase analisada. Os seguintes elementos relevantes são considerados: chunks verbais, tokens S e tokens E. São geradas duas informações para cada elemento relevante: um indicador de presença binário e o número de ocorrências em um determinado trecho da frase.

Para os classificadores de tokens S, de tokens E e o ETL-pTk, as informações adicionais são geradas da mesma maneira. Para cada token, 12 (doze) informações adicionais são geradas: seis para o trecho antes do token e seis para o trecho após o token. Para os classificadores ETL-Pair e ETL-Struc, as informações adicionais são geradas de outra maneira. Para cada par de tokens  $S$ - $E$ , são geradas 18 (dezoito) informações adicionais: seis para

o trecho *antes do token S*, seis para o trecho *após o token E* e seis para o trecho *entre o par de tokens*.

Não é demais dizer que estas informações adicionais são utilizadas apenas quando os devidos elementos relevantes estão disponíveis. Por exemplo, na classificação de tokens *S*, apenas as informações relativas ao *chunks* verbais são utilizadas. Nesta etapa as informações de tokens *S* e tokens *E* ainda não estão disponíveis.

### 3.4 *Baseline System*

O ETL necessita de um classificador inicial, dado que ele é um algoritmo baseado em correção de erros. Este classificador inicial é chamado *Baseline System* (BLS). O BLS utilizado neste trabalho é aquele proposto na CoNLL'2001, que consiste em delimitar orações como sendo as próprias frases.

## 4 Modelagem como Otimização Combinatória

As abordagens apresentadas na seção anterior para a terceira sub-tarefa são todas baseadas em classificadores. Nesta seção, uma abordagem diferente é apresentada. Seguindo a estratégia proposta nos melhores trabalhos da literatura [CM01, CMPR02, CMC05], a terceira sub-tarefa é modelada como um problema de otimização combinatória.

Dada uma frase e um conjunto de orações  $C = \{x \mid x = (x_s, x_e)\}$ , onde  $x_s$  é a posição do token inicial (token *S*) da oração  $x$  e  $x_e$  é a posição do token final (token *E*),  $C$  é dito *consistente* se, e somente se, respeitar as seguintes restrições:

- Para toda oração  $x \in C$ , o token inicial deve vir antes ou na mesma posição do token final ( $x_s \leq x_e, \forall x \in C$ ).
- Para quaisquer duas orações candidatas  $x, y \in C$ , uma das seguintes situações é verdadeira (assumindo, sem perda de generalidade, que  $x_s \leq y_s$ ):
  - As duas orações estão disjuntas ( $x_s \leq x_e < y_s \leq y_e$ ).
  - As duas orações estão aninhadas ( $x_s \leq y_s \leq y_e \leq x_e$ ).

A abordagem da terceira sub-tarefa como um problema de otimização combinatória consiste no seguinte. Dados um conjunto de orações candidatas  $C$  de uma frase e uma *função de avaliação* de orações  $s : C \rightarrow \mathbb{R}$ , a

solução do problema é o conjunto *ótimo*  $C^* \subseteq C$ , tal que  $C^*$  é consistente e  $s(C^*)$  é máximo, onde  $s(C^*) = \sum_{x \in C^*} s(x)$ .

Portanto, esta abordagem consiste em encontrar o melhor conjunto consistente de orações, dentre um dado conjunto de orações candidatas. Neste trabalho, o conjunto de orações candidatas é composto por todos os pares possíveis de tokens  $S$  e tokens  $E$ , identificados nas duas primeiras sub-tarefas. Diferentes funções de avaliação de uma oração são testadas.

Encontrar o conjunto ótimo de orações  $C^* \subseteq C$  é um problema de otimização combinatória. Seja  $C_{i,j}^* \subseteq C$  o conjunto ótimo de orações para a subsequência da frase em questão que inicia no token  $i$  e termina no token  $j$ , onde  $1 \leq i, j \leq n$  e  $n$  é a quantidade de tokens da frase. Então,  $C_{i,j}^*$  pode ser definido recursivamente como:

$$C_{i,j}^* = \operatorname{argmax}_{i < i' \leq j} \{s(C_{i,i'-1}^* \cup C_{i',j}^* \cup \{(i, j)\})\},$$

caso o token  $i$  seja classificado como  $S$  e o token  $j$  como  $E$ , ou

$$C_{i,j}^* = \operatorname{argmax}_{i < i' \leq j} \{s(C_{i,i'-1}^* \cup C_{i',j}^*)\},$$

caso contrário. Desta forma,  $C^* = C_{1,n}^*$ .

A resolução recursiva deste problema tem custo exponencial no tamanho da frase. Porém, este problema pode ser resolvido através de programação dinâmica. O passo base do processo consiste em construir  $C_{i,i}^*$ ,  $\forall 1 \leq i \leq n$ , ou seja, todos as subsequências de tamanho 1. Em cada passo  $k$  subsequente, onde  $0 < k \leq n$ , são obtidos os conjuntos  $C_{i,i+k}^*$ ,  $\forall 1 \leq i \leq n - k$ , ou seja, as soluções para as subsequências de tamanho  $k$ . Estas solução são construídas em ordem crescente em  $k$ . Ao final deste procedimento, a solução ótima é o conjunto  $C_{1,n}^* = C^*$ .

Esta abordagem tem resultados altamente dependentes da função de avaliação  $s$  utilizada. Uma função constante  $s(x) = 1$ , por exemplo, faz com que seja maximizado o número de orações da frase. Uma estratégia mais elaborada é utilizar uma combinação das confianças dos classificadores de token  $S$  e  $E$ . A segunda abordagem obtém resultados melhores e é adotada neste trabalho. As confianças são obtidas através da versão probabilística do ETL [dSM07]. Este sistema é denominado aqui *DP*.

#### 4.1 *DP-Mult*

Existe perda de informação quando se classifica um token apenas como  $S$  e/ou  $E$ . Visando considerar a quantidade máxima de orações que um token inicia ou termina, foram criadas novas *features* de multiplicidade da seguinte

forma. Dados limites  $s_{max}$  e  $e_{max}$ , são criadas  $s_{max}$  *features* cujos valores, para a *feature*  $s_i$ , onde  $1 \leq i \leq s_{max}$ , são:

- $S$  – se, e somente se, o token associado abre pelo menos  $i$  orações ou
- $X$  – caso contrário.

A criação das *features* de encerramento de oração  $e_i$  é análoga.

De posse destas *features*, é possível modificar a programação dinâmica para considerar que o conjunto  $C_{i,j}^*$  seja aquele no qual os tokens  $i$  e  $j$  limitam pelo menos  $0, 1, \dots, s_{max}$  e  $0, 1, \dots, e_{max}$  orações, respectivamente. Esta nova abordagem é denominada *DP-Mult*. Contudo, ela aumenta muito a dimensão da tabela de programação dinâmica.

Como, na tabela de programação dinâmica, as duas dimensões de cada intervalo  $[i, k]$  e  $[k+1, j]$  devem ser combinadas entre si, o custo do algoritmo aumenta de  $\Theta((s_{max} \times e_{max})^2)$  vezes. Mesmo para valores pequenos de  $s_{max}$  e  $e_{max}$ , portanto, o algoritmo se torna demasiadamente caro. Felizmente, no corpus de treinamento não há tokens iniciando mais do que 4 orações e nem terminando mais do que 6 orações.

## 5 Experimentos

Nesta seção são apresentados e discutidos os principais resultados obtidos com as abordagens desenvolvidas. O corpus CoNLL’2001 já está dividido em três partes: treino, desenvolvimento e teste. Para comparar as diferentes abordagens desenvolvidas neste trabalho, o corpus de *desenvolvimento* é utilizado. Para avaliar o desempenho da melhor abordagem deste trabalho em relação aos sistemas no estado da arte, o corpus de *teste* é utilizado.

### 5.1 Comparação das estratégias propostas

Um resumo do desempenho obtido por cada abordagem desenvolvida neste trabalho, no corpus de desenvolvimento da CoNLL’2001, é apresentado na Tabela 4. Nas primeiras duas linhas desta tabela encontram-se os resultados para os classificadores de tokens  $S$  e de tokens  $E$ .

O sistema *DP* atinge um valor de  $F_{\beta=1}$  mediano. A abrangência desta abordagem é alta – é o maior valor dentre os sistemas desenvolvidos. Por outro lado, a precisão é muito baixa. Isto é devido à função de avaliação de orações utilizada. Com esta função, a programação dinâmica insere muitas orações espúrias, resultantes da associação de dois tokens com confiança baixa, porém, com valor positivo. Outras formas de combinar os valores

<i>Modelo</i>	<i>Precisão</i>	<i>Abrangência</i>	$F_{\beta=1}$
Tokens <i>S</i>	94,02	90,86	92,42
Tokens <i>E</i>	89,00	88,98	88,99
<i>DP</i>	65,44	78,09	71,21
<i>DP-Mult</i>	82,99	66,45	73,80
<i>ETL-pTk</i>	82,72	76,95	79,73
<i>ETL-Pair</i>	85,57	77,48	81,33
<i>ETL-Struc</i>	87,21	77,80	82,24

Tabela 4: Resultados no corpus de *desenvolvimento* da CoNLL’2001.

de confiança dos tokens *S* e *E* foram experimentadas (utilizando limiares de confiança, por exemplo). Entretanto, os resultados não foram melhores.

O sistema *DP-Mult* alcança uma melhoria expressiva de 2,59% em relação ao sistema *DP* com relação ao  $F_{\beta=1}$ . Além disso, este sistema melhora muito a precisão do anterior. Entretanto, a abrangência é baixa.

O modelo *ETL-pTk* ilustra a aplicação direta do ETL nas três sub-tarefas de identificação de orações. O resultado deste sistema é bom, visto sua simplicidade. O modelo *ETL-Pair* consegue uma melhoria de 1,6% no  $F_{\beta=1}$ , sobre o *ETL-pTk*. Este resultado evidencia a relevância desta abordagem.

O sistema *ETL-Struc* obtém os melhores resultados dentre todas as abordagens desenvolvidas neste trabalho. Mais especificamente, este modelo consegue um incremento de 2,51%, no  $F_{\beta=1}$ , sobre o modelo *ETL-pTk*, e 0,91 sobre o modelo *ETL-Pair*. Este resultado mostra que o ETL consegue aprender alguma informação sobre a dependência entre as orações candidatas em uma mesma frase.

## 5.2 Comparação com o estado da arte

Na Tabela 5 são apresentados os resultados – relativos ao corpus de *teste* da CoNLL’2001 – do modelo *ETL-Struc*, do melhor modelo publicado na literatura [CMC05] e dos sistemas que obtiveram, respectivamente, a primeira [CM01] e a segunda [MP01] colocação na competição da CoNLL’2001.

O modelo *ETL-Struc* obtém resultados significativamente melhores do que o segundo colocado da CoNLL’2001 [MP01]. Além disso, os resultados do modelo utilizando ETL não ficam muito distantes dos melhores resultados disponíveis na literatura.

	<i>Precisão</i>	<i>Abrangência</i>	$F_{\beta=1}$
[CMC05]	88,17	82,10	85,03
[CM01]	90,18	78,11	83,71
<b><i>ETL-Struc</i></b>	<b>86,37</b>	<b>75,45</b>	<b>80,55</b>
[MP01]	70,85	70,51	70,68

Tabela 5: Resultados no corpus de *teste* da CoNLL’2001.

## 6 Conclusão

Neste trabalho são apresentadas diversas abordagens para a importante tarefa de identificação de orações. Estas abordagens são baseadas no algoritmo de aprendizado de máquina ETL. Os sistemas desenvolvidos são avaliados no corpus da CoNLL’2001 [SD01], que é composto por textos em língua inglesa retirados do *Wall Street Journal*. Os resultados obtidos são comparados com o estado da arte na área [CM01, CMPR02, CMC05]. A contribuição das diferentes abordagens desenvolvidas são enfatizadas através de comparações de desempenho no corpus CoNLL’2001.

Os resultados alcançados pelo melhor sistema desenvolvido são competitivos com o estado da arte. Além disso, a modelagem utilizada neste trabalho é simples e geral, permitindo sua aplicação em outros problemas de linguística computacional. As modelagens utilizadas pelos melhores sistemas da literatura [CM01, CMC05] são bem complexas e utilizam características específicas da tarefa de identificação de orações. Por outro lado, algumas técnicas de modelagem utilizadas em [CMC05] são muito poderosas e podem ser aproveitadas para estender os modelos ETL.

Apesar de um corpus em língua inglesa ser utilizado neste trabalho, as técnicas desenvolvidas são gerais o suficiente para serem aplicadas em outras línguas. O projeto *Floresta Sintá(c)tica* [FRB08] mantém um corpus em língua portuguesa com aproximadamente 6,7 milhões de palavras. Este corpus possui anotações – geradas automaticamente pelo analisador *Palavras* [Bic00] – com a análise completa de constituintes. Um dos tipos de informação contido neste corpus informa sobre as orações do texto. Como uma parte deste corpus – o *Bosque* – foi verificado manualmente, é possível utilizá-lo em algoritmos de aprendizado supervisionado, como o ETL.

## Referências

- [Bic00] Eckhard Bick. *The Parsing System “Palavras”: Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework*. PhD thesis, Aarhus University, Aarhus, Denmark, 2000.
- [Bri95] Eric Brill. Transformation-based error-driven learning and natural language processing: a case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565, 1995.
- [CM01] Xavier Carreras and Lluís Màrquez. Boosting trees for clause splitting. In *Proceedings of Fifth Conference on Computational Natural Language Learning*, Toulouse, France, 2001.
- [CMC05] Xavier Carreras, Lluís Màrquez, and Jorge Castro. Filtering-ranking perceptron learning for partial parsing. *Machine Learning*, 60(1–3):41–71, 2005.
- [CMPR02] Xavier Carreras, Lluís Màrquez, V Punyakanok, and D Roth. Learning and inference for clause identification. In *Proceedings of the Thirteenth European Conference on Machine Learning*, pages 35–47, 2002.
- [dSM07] Cícero N. dos Santos and Ruy L. Milidiú. Probabilistic classifications with TBL. In *Computational Linguistics and Intelligent Processing*, volume 4394 of *Lecture Notes in Computer Science*, pages 196–207. Springer, 2007.
- [dSM09] Cícero N. dos Santos and Ruy L. Milidiú. *Foundations of Computational Intelligence, Volume 1: Learning and Approximation*, volume 201 of *Studies in Computational Intelligence*, chapter Entropy Guided Transformation Learning, pages 159–184. Springer, 2009.
- [dSMR08] Cícero N. dos Santos, Ruy Luiz Milidiú, and Raúl P. Renteria. Portuguese part-of-speech tagging using entropy guided transformation learning. In *Proceedings of PROPOR 2008*, Aveiro, Portugal, 2008.
- [FRB08] Cláudia Freitas, Paulo Rocha, and Eckhard Bick. Floresta Sintá(c)tica: Bigger, thicker and easier. In António Teixeira, Vera Lúcia Strube de Lima, Luís Caldas de Oliveira, and Paulo

Quaresma, editors, *Computational Processing of the Portuguese Language*, volume 5190 of *Lecture Notes in Computer Science*, pages 216–219. Springer, 2008.

- [MCLS08] Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159, 2008.
- [MdSD08a] Ruy L. Milidiú, Cícero N. dos Santos, and Julio C. Duarte. Phrase chunking using entropy guided transformation learning. In *Proceedings of ACL-08: HLT*, pages 647–655, Columbus, USA, 2008. Association for Computational Linguistics.
- [MdSD08b] Ruy L. Milidiú, Cícero N. dos Santos, and Júlio C. Duarte. Portuguese corpus-based learning using ETL. *Journal of the Brazilian Computer Society*, 14(4), 2008.
- [MP01] Antonio Molina and Ferran Pla. Clause detection using HMM. In *Proceedings of CoNLL-2001*, Toulouse, France, 2001.
- [MSM93] Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.
- [Niv05] Joakim Nivre. Dependency grammar and dependency parsing. Technical report, Växjö University: School of Mathematics and Systems Engineering, 2005.
- [San00] Erik F. T. K. Sang. Text chunking by system combination. In *Proceedings of Conference on Computational Natural Language Learning*, Lisbon, Portugal, 2000.
- [SD01] Erik F. T. K. Sang and Hervé Déjean. Introduction to the CoNLL-2001 shared task: Clause identification. In *Proceedings of Fifth Conference on Computational Natural Language Learning*, Toulouse, France, 2001.