Dynamic Programming on Tensors for Solving the Problem of Dependency Parsing in NLP

Optimization&NLA course project

Sergey Divakov, Anastasia Koloskova, Alfredo De la Fuente and Vladislav Pimanov

Skolkovo Institute of Science and Technology, Data Science Department
Moscow, Russia

22nd December 2017

Skoltech
Skolkovo Institute of Science and Technology

Outline

Introduction

- Syntactic dependency parsing is the important problem in statistical natural language processing.
- Our goal was to speed up the parsing process.
- Using tensor formulation of the inside-outside algorithm we compared different tensor decompositions.
- Tucker and Tensor Train decompositions were applied.
- We were able to achieve significant performance increase with good accuracy results.

Skoltech
Skolkovo Institute of Science and Technology

Setup

Probabilistic Context-free Grammars

$$G = (\mathcal{N}, \mathcal{L}, \mathcal{R}, \mathcal{P}, \pi),$$

- $\mathcal{N}$ — nonterminal symbols.
- $\mathcal{L}$ — words (lexical tokens).
- $\mathcal{R}$ — set of rules: $a \to bc$ or $a \to x$, $a, b, c \in \mathcal{N}$, $x \in \mathcal{L}$.
- $\mathcal{P}$ — transition probabilities $p(a \to bc|a)$ and $p(a \to x|a)$.
- π_a — probability of a being the root symbol.
- All probabilities satisfy normalization conditions.

Setup

Our purpose: given a sentence find the most probable tree of this sentence.

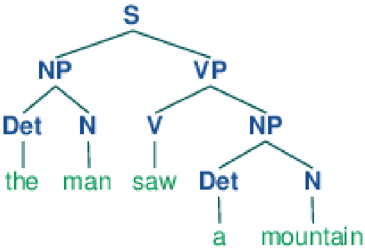

Figure: Parse tree representation of *the man saw a mountain*

Algorithms

Inside-Outside

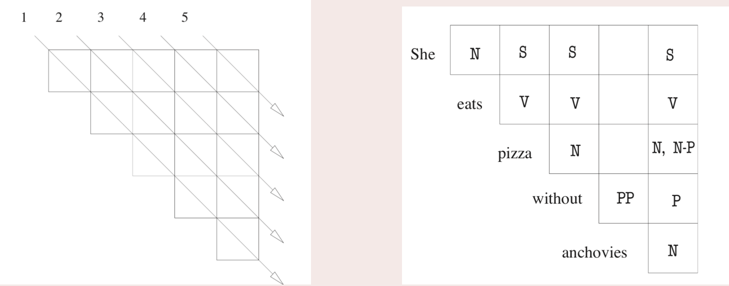

Figure: Inside Recursion

Skoltech
Skolkovo Institute of Science and Technology

Algorithms

Algorithm 1 Inside-Outside Algorithm in the Tensor Form

1: **(Inside base case):** $\forall (a \to x_i) \in \mathcal{R}\ [\alpha^{i,i}]_a = p(a \to x|a)$

2: **(Inside recursion):**

$$[\alpha^{i,j}]_a = \sum_{k=i}^{j-1} \sum_{a \to bc} T^{a \to bc}([\alpha^{i,k}]_b, [\alpha^{k+1,j}]_c)$$

3: **(Outside base case):** $\forall a \in \mathcal{N}\ [\beta^{1,N}]_a = \pi_a$

4: **(Outside recursion):**

$$[\beta^{i,j}]_a = \sum_{k=1}^{i-1} \sum_{b \to ca} T_{(1,2)}^{b \to ca}([\beta^{k,j}]_b, [\alpha^{k,i-1}]_c) + \\ + \sum_{k=j+1}^{N} \sum_{b \to ac} T_{(1,3)}^{b \to ac}([\beta^{ik}]_b, [\alpha^{j+1,k}]_c)$$

5: **(Marginals):** $\mu(a, i, j) = [\alpha^{ij}]_a \cdot [\beta^{ij}]_a$

Skoltech
Skolkovo Institute of Science and Technology

Algorithms

Algorithm 2 The Labelled Recall Algorithm from Goodman

1: **(Marginals):** Compute $\mu(a, i, j)$ using inside-outside.
Finding the most probable tree using dynamic programming:

2: **(Base case):**

$$\gamma^{ii} = \max_{(a \to x_i) \in \mathcal{R}} \mu(a, i, i)$$

3: **(Maximize Labelled Recall):**

$$\gamma^{i,j} = \max_{a \in \mathcal{N}} \mu(a, i, j) + \max_{i \le k < j} (\gamma^{i,k} + \gamma^{k+1,j})$$

Skoltech
Skolkovo Institute of Science and Technology

Tensor Decompositions (3D case)

Canonical

$$\mathcal{A} = \sum_{i=1}^{r} \lambda_i \mathbf{a}_i^1 \otimes \mathbf{a}_i^2 \otimes \mathbf{a}_i^3,$$

$\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $\lambda_i \in \mathbb{R}$ and $\mathbf{a}_i^j \in \mathbb{R}^{n_j}$

Time of multiplying by vector (of size n_3): $\mathcal{O}(rn_3)$

Tensor Train

$$\mathcal{A}(i_1, i_2, i_3) = \sum_{k_1=1}^{r_1} \sum_{k_2=1}^{r_2} G_1(i_1, k_1) G_2(k_1, i_2, k_2) G_3(k_2, i_3),$$

$\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, $G_1 \in \mathbb{R}^{n_1 \times r_1}$, $G_2 \in \mathbb{R}^{r_1 \times n_2 \times r_2}$, $G_3 \in R^{r_2 \times n_3}$.

Time of multiplying by vector (of size n_3): $\mathcal{O}(r_1 r_2 n_3)$

Skoltech
Skolkovo Institute of Science and Technology

Tensor Decompositions (3D case)

Tucker

$$\mathrm{vec}(\mathcal{A}) = (W \otimes V \otimes U) \cdot \mathrm{vec}(\mathcal{C}),$$

$\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}, U \in \mathbb{R}^{n_1 \times r_1}$, $V \in \mathbb{R}^{n_2 \times r_2}$, $W \in \mathbb{R}^{n_3 \times r_3}$ and $\mathcal{C} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$

Time of multiplying by vector (of size n_3):
$\mathcal{O}\left(n_3 r_3 + r_1 r_2 r_3 + r_1 r_2 n_1 + r_2 n_2\right) = \mathcal{O}(nr^2 + r^3)$.

Experiments

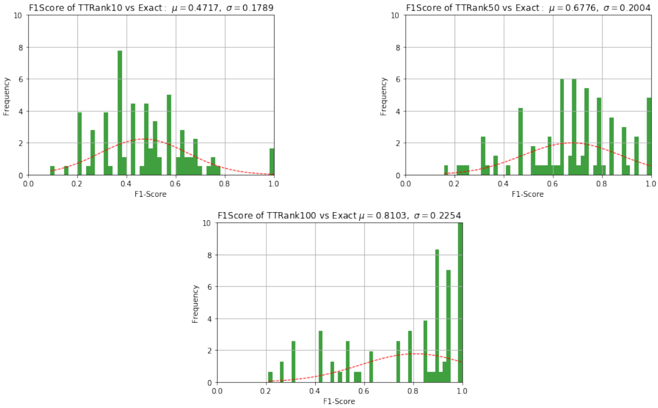

Figure: F1 scores TT

Skoltech
Skolkovo Institute of Science and Technology

Experiments

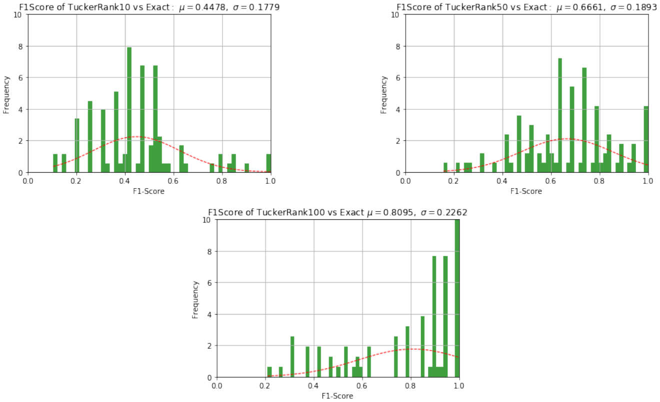

Figure: F1 scores Tucker

Experiments

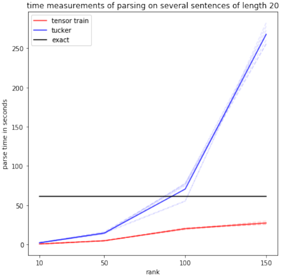

Figure: Time efficiency

Skoltech
Skolkovo Institute of Science and Technology

Experiments

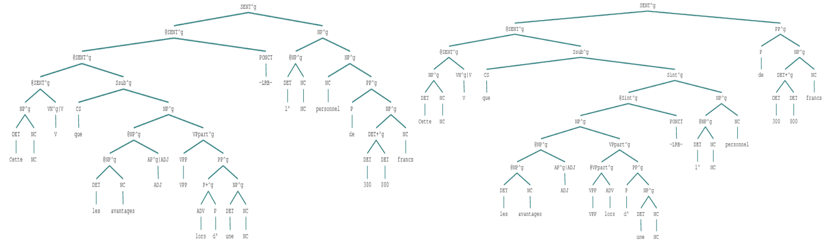

Figure: Tree comparison

Conclusion

- Using tensor formulation of the inside-outside algorithm we tried different tensor decompositions to speed up parsing process.
- Tucker and Tensor Train decompositions were applied.
- Experiments showed that using tensor decompositions we can achieve significant speed up with the good accuracy of parsing.