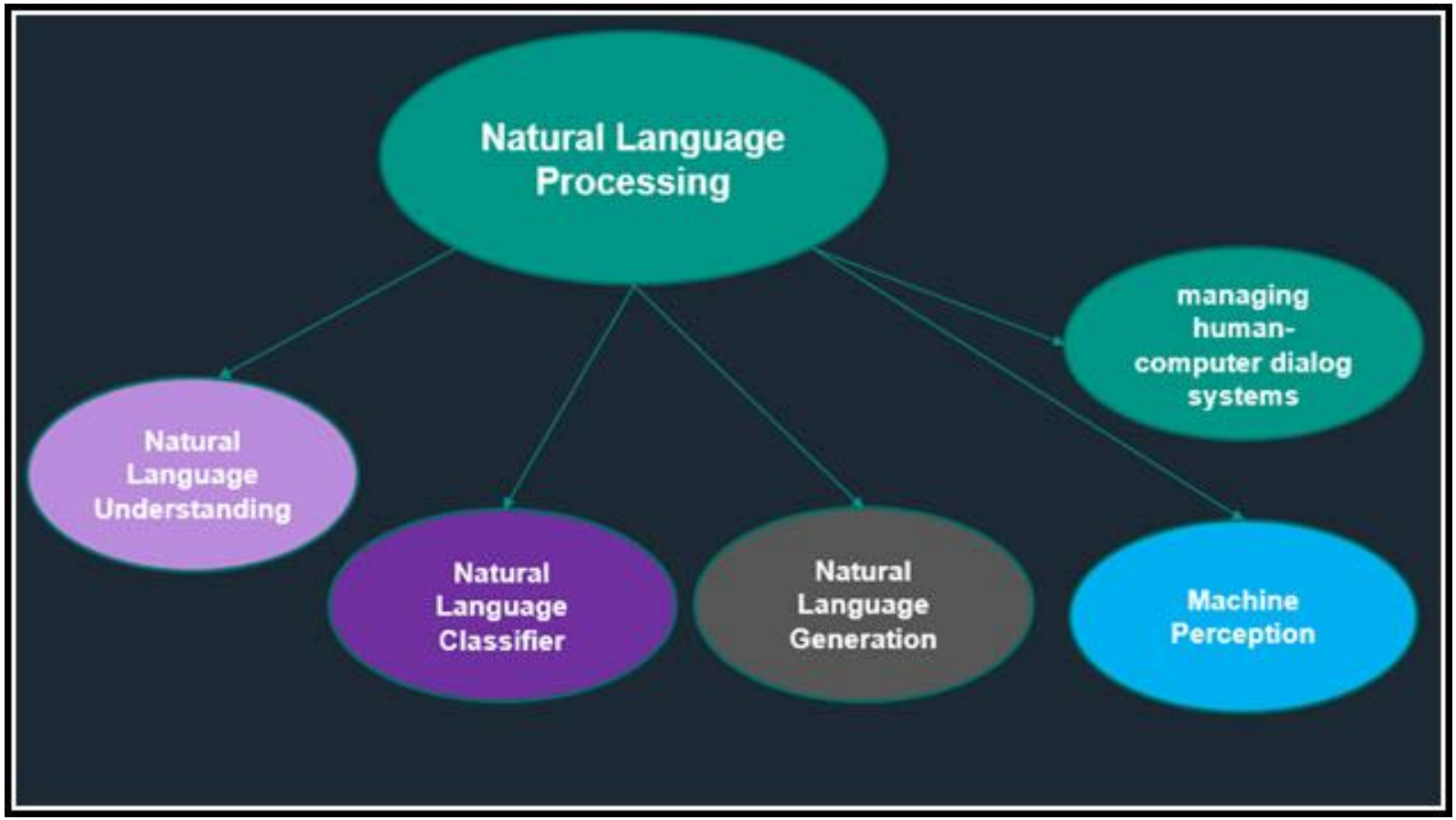
Deep Learning (Parte 4)

Índice Geral

1 Introdução ao Processamento de Linguagem Natural

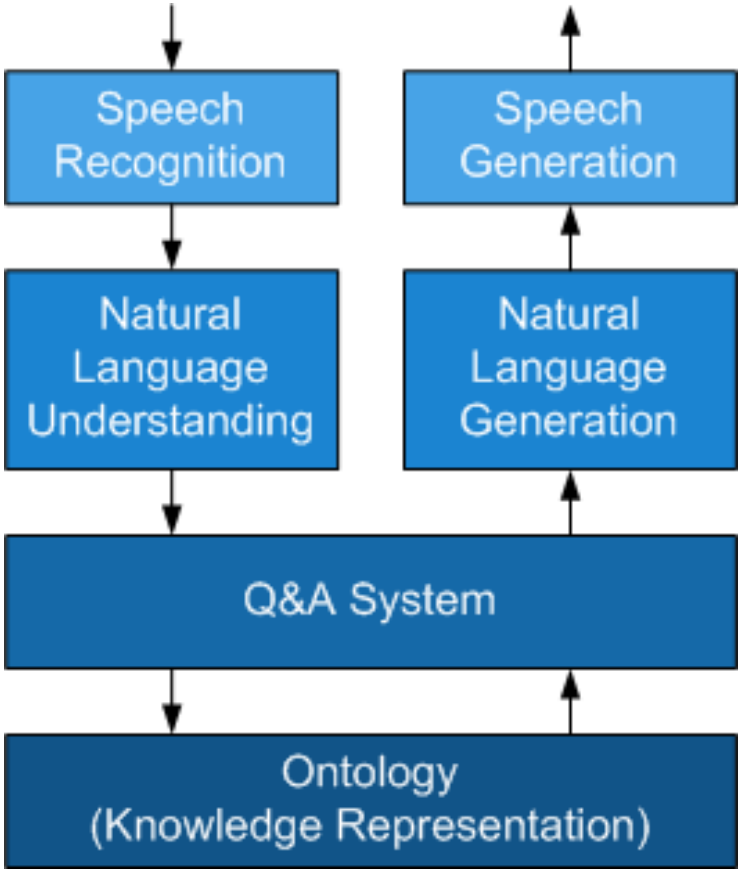
- Parte deste material é baseado na monografia de Trabalho de Conclusão de Curso de Graduação de Brenda Oliveira Ramirez, intitulado "Extração de atributos por codificação densa em Processamento de Linguagem Natural", Faculdade de Engenharia Elétrica e de Computação (FEEC/Unicamp), 2017.
- Parte deste material foi extraído do tutorial de autoria de SOCHER & MANNING (2013), disponível em [https://nlp.stanford.edu/courses/NAACL2013/].
- Processamento de linguagem natural (PLN ou NLP, do inglês *Natural Language Processing*) envolve o tratamento computacional da linguagem humana.
- No contexto de aprendizado de máquina, o objetivo é fazer com que os computadores entendam, de alguma forma, a linguagem humana e a reproduzam.
- Recentemente, tem havido uma ampla difusão de aplicações bem-sucedidas na forma de máquinas de busca para conteúdo textual, como Google, Yahoo!, Bing e Baidu.

- Também temos máquinas capazes de responder perguntas, como é o caso do Watson da IBM e do aplicativo SIRI, exclusivo da Apple, que também faz recomendações e executa comandos.
- Há também uma acentuada demanda por técnicas de sumarização de textos, como as propostas em PAULUS et al. (2017). Cabe destaque também para a análise de sentimento, como na proposta de IRSOY & CARDIE (2014).
- O sistema de tradução Google Translate tem evoluído muito em termos de desempenho e é outro importante exemplo de sucesso no processamento de linguagem natural.
- Já mais consolidados e possivelmente menos relevantes do que já foram tempos atrás (em virtude da expansão das redes sociais), temos os agregadores de notícias, que são agentes virtuais que visitam muitos sites diferentes com frequência para descobrir se o conteúdo do site foi atualizado. A tecnologia de agregação ajuda a consolidar muitos sites em uma única página.



Fonte: https://www.ibm.com/developerworks/community/blogs/5things/entry/5_things_to_know_about_Watson_Natural_Language_Classifier?lang=en

Fonte: https://toxsl.com/service/57/natural-language-processing



Fonte: https://developer.ibm.com/articles/a-beginners-guide-to-natural-language-processing/

- Ontologia, em PLN e em outras áreas da computação, se refere ao conhecimento de domínio (*domain knowledge*).
- Ontologia envolve taxonomia, atributos, relações, regras e eventos.

- "O Processamento de Linguagem Natural (PLN, ou NLP, do inglês *Natural Language Processing*) descreve um ramo da Inteligência Artificial (IA) que automatiza a compreensão e a síntese de linguagem para que os computadores e os seres humanos possam se comunicar sem intermediários. Para interagir com humanos, os computadores devem adotar e compreender sintaxe (gramática), semântica (significado da palavra), morfologia (e.g. passado, presente, futuro) e pragmática (conversação)."

  Fonte: https://www.dataversity.net/what-is-natural-language-processing-nlp/

- São tarefas que se mostraram bastante complexas, impondo muitos desafios para serem adequadamente assimiladas por uma máquina.
- Além disso, o PLN envolve técnicas multidisciplinares, fundamentadas em conceitos de linguística, ciência da computação, matemática, estatística, inteligência artificial, dentre outras áreas.

- Aspectos sintáticos e semânticos estão envolvidos, além da intenção do falante (por exemplo, compartilhar conhecimento ou crenças) e da capacidade de percepção e compreensão do ouvinte, ambas vinculadas a um contexto em que a linguagem está sendo empregada.

Fonte: https://www.shutterstock.com/pt/search/listening+reading+speaking+writing

- Para o desenvolvimento da Inteligência Artificial, é fundamental que as máquinas tenham capacidade de aprendizado e de comunicação a fim de permitir o aprendizado por experiência e interação, assim como acontece com os seres humanos (BARONI et al., 2017).
- As redes neurais artificiais, com destaque para as redes com arquitetura profunda, possibilitam isso através de múltiplos níveis de representação ao realizar a composição de camadas que aumentam o grau de abstração a cada nível, permitindo o aprendizado de funções mais complexas (LECUN et al., 2015).
- O domínio da linguagem humana pelas máquinas possibilitará o acesso a uma parte significativa do nosso conhecimento, pois este está codificado em linguagem natural. Além disso, cabe salientar que, de formas variadas, máquinas podem compartilhar seus cérebros, algo inacessível a seres humanos.

- No entanto, aspectos como ambiguidade e finitude da linguagem criam enormes obstáculos para que a intenção do falante seja efetivamente capturada pelo ouvinte.
- A representação vetorial, tão adequada em dispositivos computacionais, tende a ser esparsa no tratamento de linguagem natural, além de apresentar dificuldade de codificação de relações semânticas e sintáticas presentes na linguagem. Por esta razão, as soluções estado-da-arte em processamento de linguagem natural tendem a operar com um espaço denso de representação, promovendo *word embedding* ou codificação densa.
- Neste espaço de codificação densa, palavras semanticamente similares tendem a ser mapeadas próximas entre si. Além disso, a translação no espaço de codificação densa também pode codificar informação. Por exemplo, o deslocamento no espaço de codificação densa para se ir do vetor ‘king’ ao vetor ‘man’, por exemplo, tende a ser bem semelhante ao deslocamento necessário para se ir do vetor ‘queen’ ao

vetor ‘woman’. Alguns resultados de codificação permitiram até usar a composição para produzir novos significados: por exemplo, a soma do vetor ‘Germany’ e do vetor ‘capital’, produzem um resultado semelhante à representação de ‘Berlin’ (MIKOLOV et al., 2013a; 2013b).

- As principais ideias para gerar *word embeddings* baseiam-se na Hipótese Distribucional do linguista inglês J.R. Firsth, que afirmava que itens com distribuições similares tendem a ter significados semelhantes. Isso é resumido na sua famosa frase: “você conhecerá uma palavra pela sua companhia”.
- Os métodos distribucionais comprimem as informações de co-ocorrência de palavras em um dado contexto para obter uma representação. Um dos métodos mais relevantes é o *Latent Semantic Analysis* (LSA), que aplica decomposição em valores singulares em uma matriz contendo a contagem de palavras por documento, reduzindo a dimensão da matriz e preservando a similaridade presente na matriz original (DEERWESTER et al., 1990; LANDAUER et al., 1998).

- Modelos neurais também são capazes de extrair relações entre palavras baseado somente na informação distribucional. Nesses modelos, o contexto usualmente são as palavras próximas em uma frase. Um dos principais modelos nessa linha é o Modelo Neural e Probabilístico de Linguagem (*Neural Probabilistic Language Model*, NPLM) (BENGIO et al., 2003), que utiliza uma rede neural a fim de prever a próxima palavra, dado um contexto de tamanho fixo. Foi um dos pioneiros no uso de redes neurais para PLN com *word embedding* e motivou arquiteturas posteriores, que superaram em desempenho técnicas baseadas em LSA.

2 Codificação densa para o tratamento de linguagem natural

- A representação *one-hot* para linguagem natural tem uma interpretação negativa e outra positiva:
    - ✓ O aspecto negativo é tornar todas as palavras de uma linguagem, vistas como símbolos atômicos, equidistantes entre si, impedindo qualquer

representação semântica ou sintática. Além disso, trata-se de uma representação esparsa: por exemplo, numa linguagem com 2.500 palavras ou símbolos distintos, cada palavra ou símbolo da linguagem é representada por um vetor com 2.499 zeros e apenas 1 elemento unitário.

- ✓ O aspecto positivo também tem a ver com a equidistância entre as palavras, pois essa condição pode ser explorada na síntese de mapeamentos capazes de estabelecer relações semânticas ou sintáticas entre as palavras ou símbolos da linguagem, no sentido, por exemplo, de mapear próximas entre si, no espaço de destino, palavras que operam como sinônimos. É equivalente ao que se faz ao inicializar os pesos sinápticos de uma rede neural rasa: por não saber como o mapeamento a ser sintetizado deve se contorcer, a proposta é iniciar os pesos de modo a produzir algo próximo a um hiperplano, ou seja, um mapeamento sem contorção, que deve ser contorcido durante o treinamento da rede neural.

- A esparsidade da representação *one-hot* pode ser eliminada ao se propor uma redução de dimensão neste mapeamento, para um espaço de dimensão apropriada. De todo o conteúdo deste tópico, já deve ter ficado claro que o mapeamento recebe como entrada um vetor *one-hot* (ou um vetor com alguns poucos elementos não-nulos em aplicações que operam com múltiplas palavras de entrada) e produz na saída um ponto num espaço de características, também denominado de espaço de codificação densa ou *word embedding*.
- Este mapeamento pode ser bastante complexo, o que justifica a adoção de *deep learning* em processamento de linguagem natural. Mas dependendo da aplicação e buscando escalabilidade, também já foram considerados mapeamentos lineares.
- Uma dimensão elevada para o espaço de codificação densa torna mapeamentos lineares suficientemente poderosos em algumas aplicações, permitindo realizar operações semânticas na linguagem com operadores de transformação afim (translação e escalamento, por exemplo) no espaço de codificação densa.

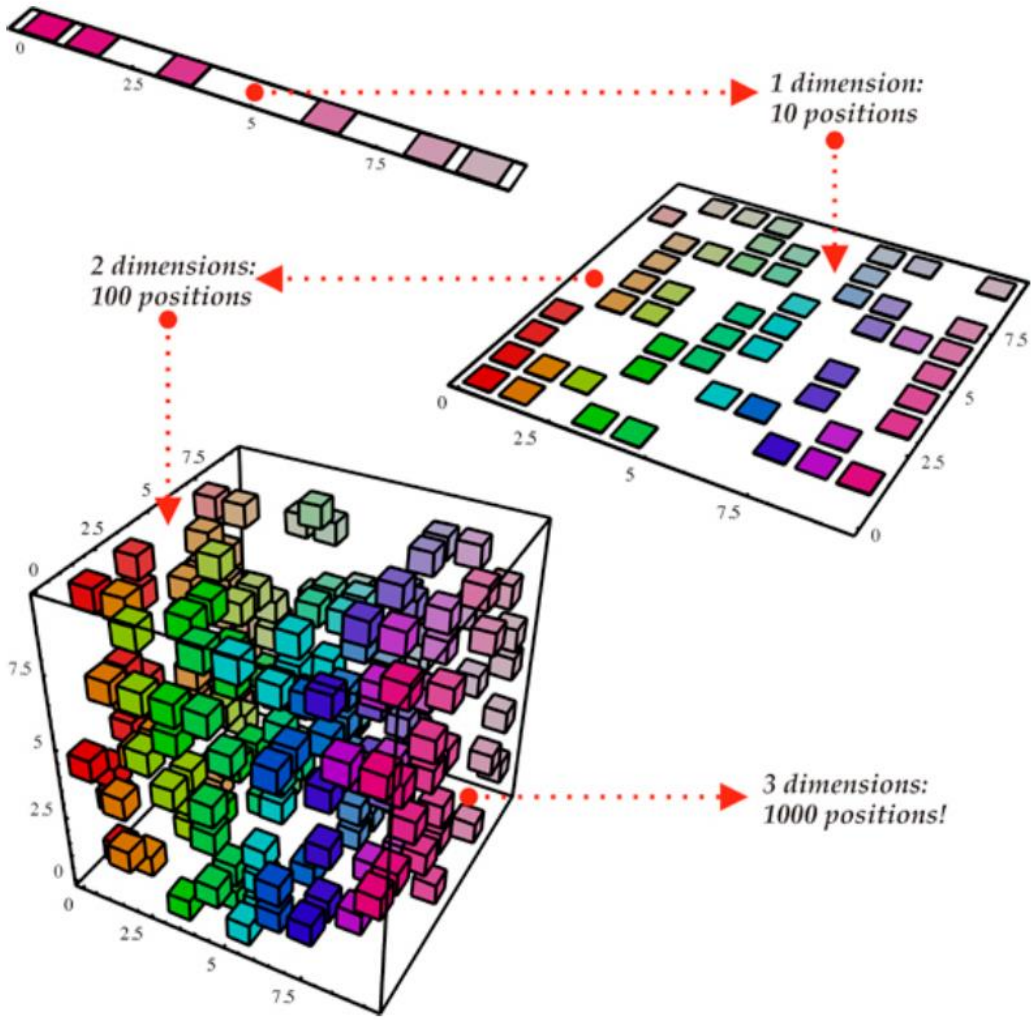


Figura 1 – Efeito do aumento de dimensão num cenário em que cada coordenada admite10 “símbolos ordenados” possíveis

- Cabe observar que, em espaços de codificação densa, diferente da codificação *one-hot*, passa a existir uma relação de ordem entre os valores de cada coordenada e, com isso, pode-se voltar a falar de distância, similaridade e vizinhança entre pontos.
- Mais ainda, com o aumento no número de coordenadas, as possibilidades de múltiplos objetos neste espaço de codificação densa estabelecerem relações de vizinhança, ou de não-vizinhança, aumentam exponencialmente.

3 Word2vec

- Uma proposta de mapeamento linear é o word2vec (MIKOLOV et al., 2013a) e suas adaptações e extensões, como em MIKOLOV et al. (2013b), que se mostram bem escaláveis. Foram utilizados bilhões de registros para se treinar uma estrutura do tipo codificador-decodificador, sendo que o gargalo tem dimensão de 50 a 300 unidades.

- As regularidades construídas automaticamente no espaço de codificação densa produzido levaram a associações do tipo: o vetor ‘Rome’ é o resultado da operação entre vetores: ‘Paris’ – ‘France’ + ‘Italy’.
- Essencialmente o word2vec é uma rede neural MLP com apenas uma camada intermediária e função de ativação identidade para todos os neurônios. A operação linguística de entrada-saída que a rede neural aprende pode ser variada. Exemplos são previsão de próxima palavra e de palavra central. O sucesso na realização dessas tarefas linguísticas requer a concepção de um espaço de codificação densa que carrega semântica e/ou sintaxe da linguagem.
- As figuras a seguir ilustram a operação. Quando aparecem múltiplos vetores de entrada, como no modelo *Continuous Bag of Words* (CBOW), ou múltiplos vetores de saída, como no modelo *Skip-Gram*, a matriz que aparece em multiplicidade é a mesma.

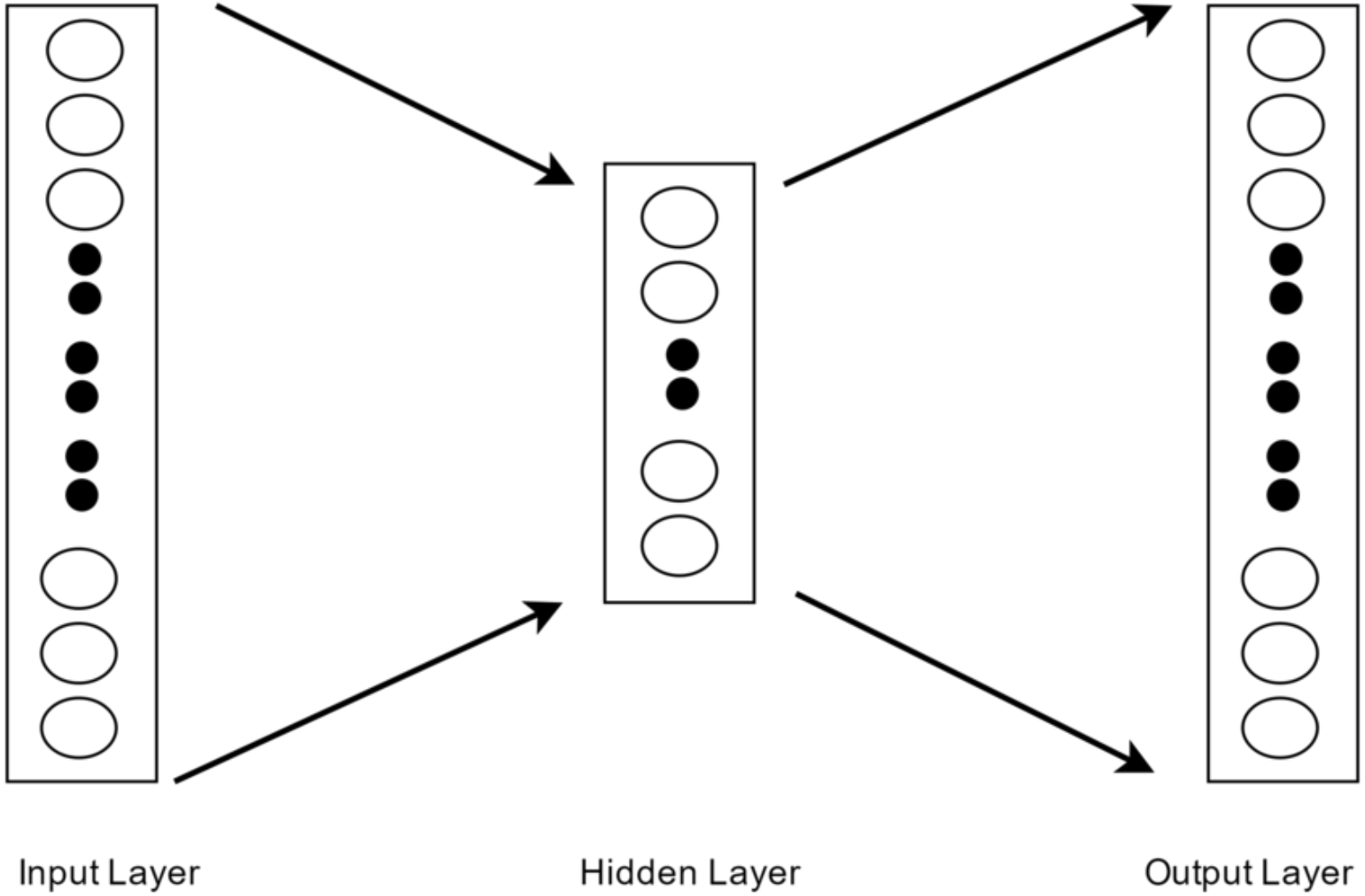


Figura 2 – Arquitetura adotada para o word2vec, com neurônios apresentando funções de ativação lineares

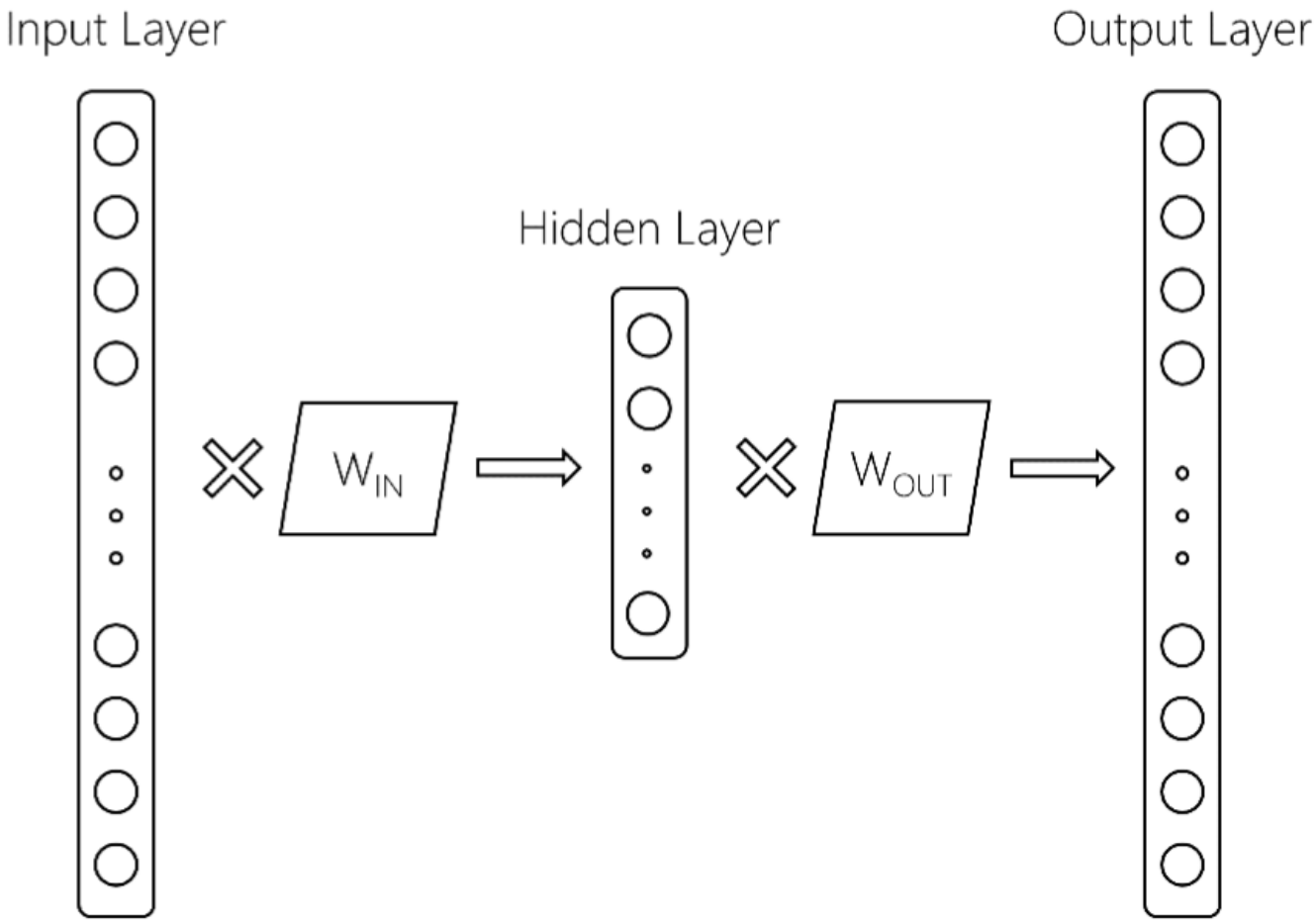


Figura 3 – Outra perspectiva da mesma arquitetura de rede para o word2vec. As matrizes $\mathbf{W_{IN}}$ e $\mathbf{W_{OUT}}$ podem ser forçadas a serem simétricas entre si, dependendo da aplicação.

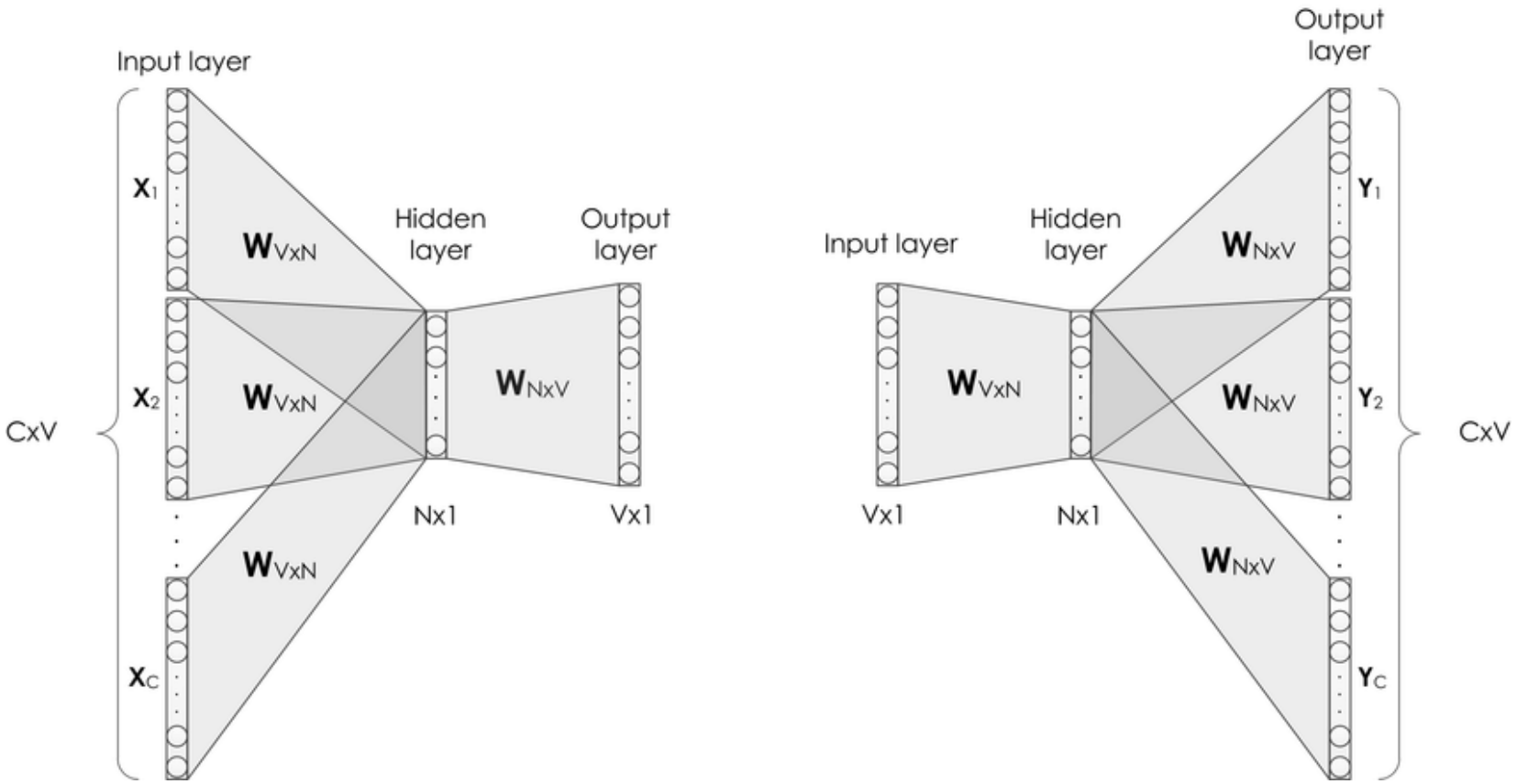


Figura 4 – À esquerda tem-se o modelo CBOW e à direita o Skip-Gram.

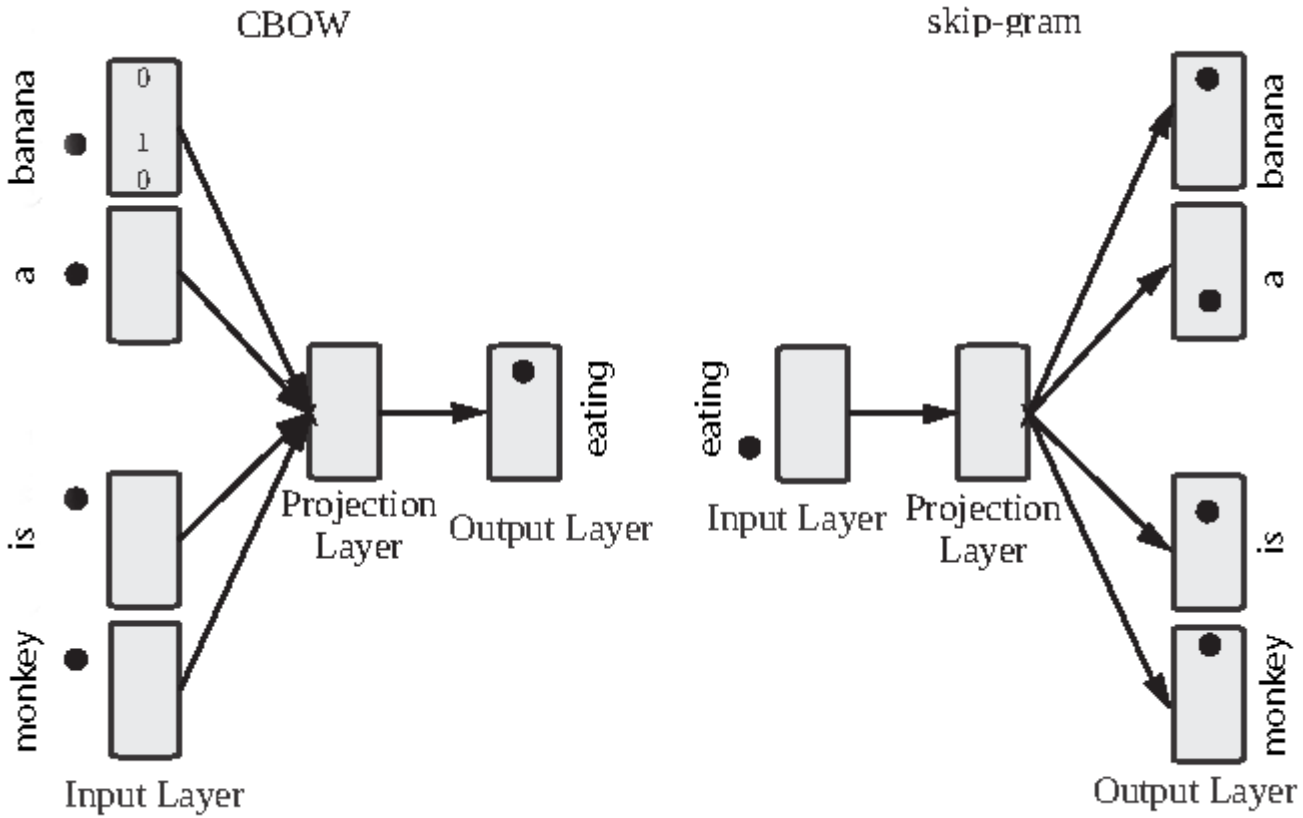


Figura 5 – Exemplo de resultado produzido por CBOW (à esquerda) e Skip-Gram (à direita).

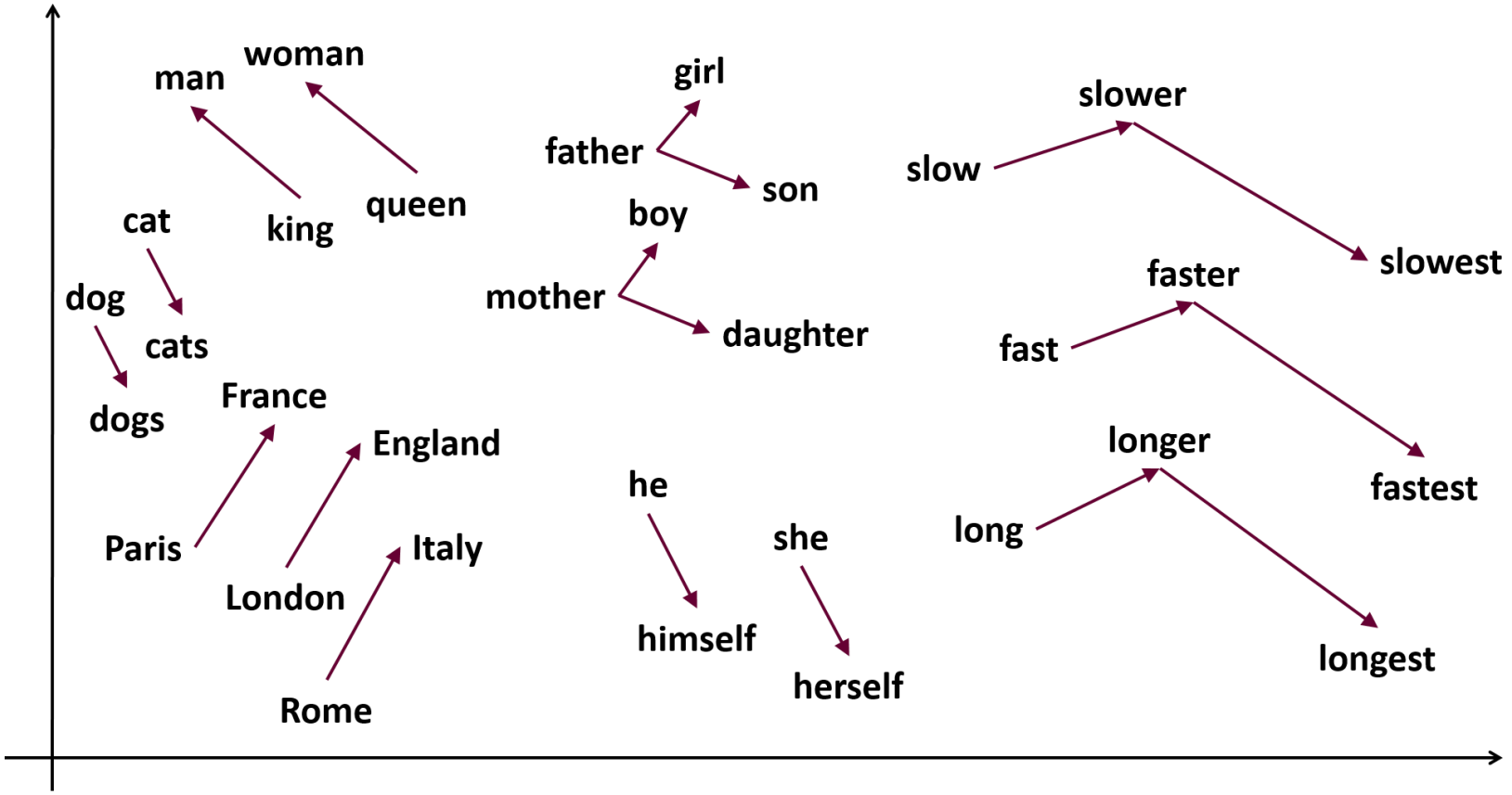


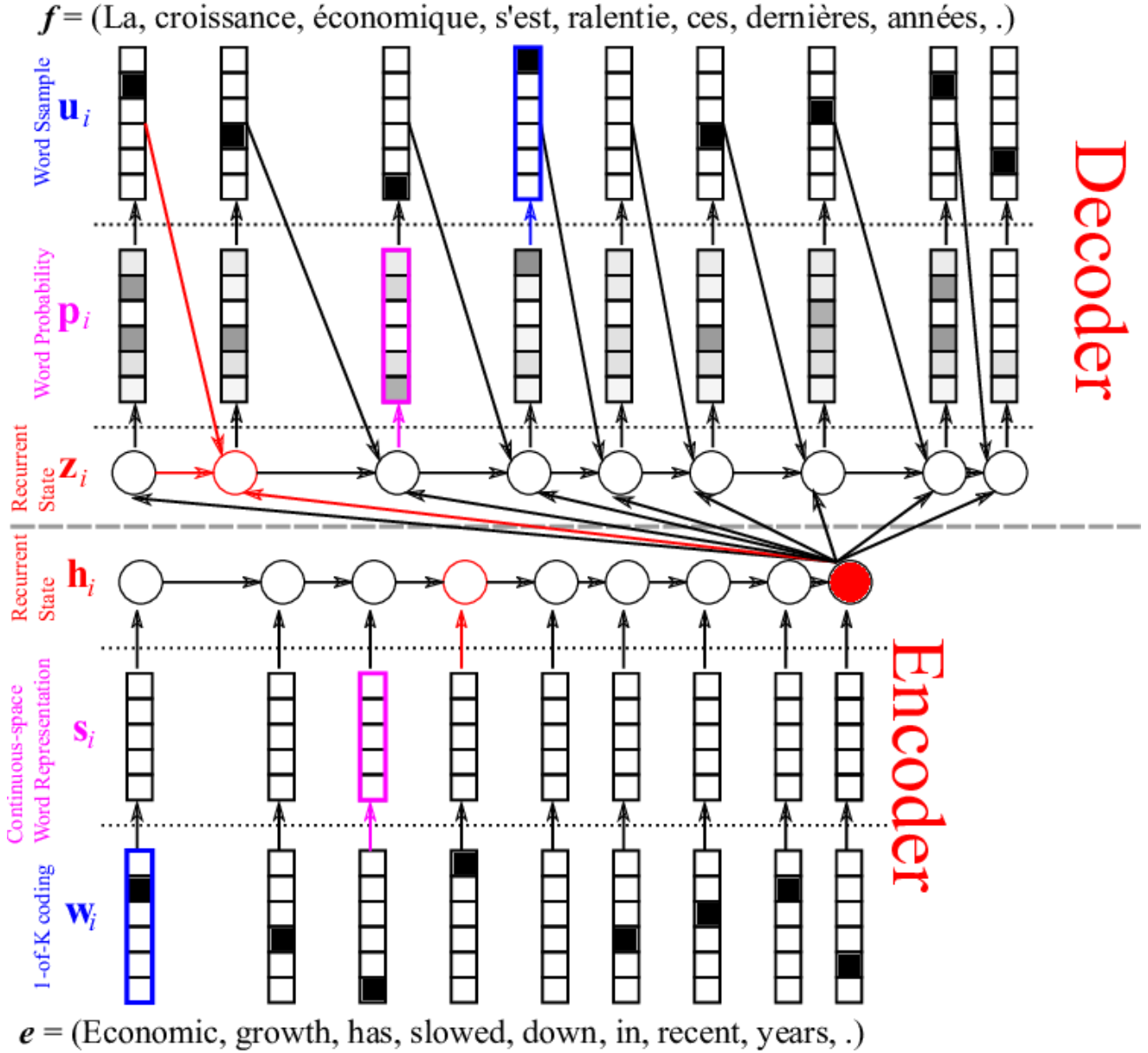
Figura 6 – Projeção 2D de alguns vetores do espaço de codificação densa produzido por modelos word2vec.

4 Máquinas tradutoras de frases

- Esta seção contém material extraído de: https://devblogs.nvidia.com/introduction-neural-machine-translation-with-gpus/
- A parte conceitual está predominantemente fundamentada em CHO et al. (2014).
- O modelo a ser adotado aqui é o de codificador-decodificador, contendo dois módulos recorrentes. A entrada é uma frase numa certa linguagem e a saída é uma proposta de tradução para a frase de entrada. Um elenco de pares de frases deve ser utilizado para guiar o processo de treinamento deste modelo.
- O primeiro módulo recorrente, na saída do codificador, é responsável por sumarizar sequências de entrada de comprimento variável, já convertidas de *one-hot* para uma codificação densa. Essa sumarização se dá na forma de um vetor de comprimento fixo, que nada mais é do que o vetor de estados deste primeiro módulo recorrente. Este vetor de estados é atualizado sequencialmente, com cada palavra da frase de entrada, em sua codificação densa.

- Já o segundo módulo recorrente é o primeiro elemento do decodificador. Com base em (1) seu estado interno, (2) última palavra definida para a frase de saída (quando houver) e (3) saída do codificador, este módulo recorrente propõe sequencialmente vetores de probabilidade, que guiam o processo de amostragem da sequência de símbolos de saída. Portanto, a saída do decodificador será a frase traduzida, também de comprimento variável.
- O critério de desempenho é maximizar a probabilidade condicional da frase esperada na saída (sequência-alvo) da máquina tradutora, dada a proposta de frase de entrada (sequência-fonte), considerando todo o conjunto de treinamento.
- Não será dada ênfase aqui às estratégias de treinamento do modelo, mas sim à sua funcionalidade e à estrutura do modelo de aprendizado. Mas cabe comentar que codificador e decodificador serão treinados conjuntamente para maximizar a probabilidade da frase de saída (sequência-alvo) condicionada à ocorrência da frase de entrada (sequência-fonte), considerando todos os pares de frases.

- A rede neural, portanto, vai operar como uma máquina de tradução estatística: a frase de saída é amostrada sequencialmente, palavra a palavra, a partir da obtenção de um vetor de probabilidades que indica a chance de escolha de cada palavra candidata, o qual é usado para amostrar a palavra seguinte da sequência de saída.
- Uma análise deste modelo codificador-decodificador, realizada por CHO et al. (2014), revela que ele sintetiza um espaço de codificação densa capaz de preservar as estruturas semântica e sintática das frases.
- A figura a seguir descreve graficamente todas as etapas de codificador e decodificador, conforme a descrição textual acima. Os módulos recorrentes aparecem desdobrados no espaço, da esquerda para a direita, indicando que a chegada de palavras e a saída de palavras é sequencial.
- Em CHO et al. (2014), cada módulo recorrente é uma GRU (*Gated Recurrent Unit*), que é um bloco LSTM descrito na Parte 2 deste Tópico 8.


f = (La, croissance, économique, s'est, ralentie, ces, dernières, années, .)
Word Sample
u_i
Word Probability
p_i
Recurrent State
z_i
Decoder
Recurrent State
h_i
Continuous-space Word Representation
s_i
1-of-K coding
w_i
Encoder
e = (Economic, growth, has, slowed, down, in, recent, years, .)

4.1 Máquina de tradução estatística

- A GRU do codificador realiza a operação:

$$h_{\langle i \rangle} = f\left(h_{\langle i-1 \rangle}, s_{\langle i-1 \rangle}\right).$$

- O vetor de estados $h_{\langle i \rangle}$ usa codificação densa, ou seja, $h_{\langle i \rangle} \in \Re^M$, com M sendo uma dimensão apropriada, dependente de cada aplicação.
- Já a GRU do decodificador realiza a operação:

$$\begin{cases} z_{\langle i \rangle} = f\left(z_{\langle i-1 \rangle}, u_{\langle i-1 \rangle}, h_{\langle \text{final} \rangle}\right) \\ p_{\langle i \rangle} = \text{softmax}\left(z_{\langle i \rangle}\right) \end{cases}.$$

- Seja T o número de palavras da linguagem de saída, então o vetor $p_{\langle i \rangle}$ tem dimensão T, assim como o vetor $z_{\langle i \rangle}$, e cada elemento de $p_{\langle i \rangle}$ é dado na forma:

$$p_{\langle i \rangle, j} = \frac{\exp\left(z_{\langle i \rangle, j}\right)}{\sum_{l=1}^{T} \exp\left(z_{\langle i \rangle, l}\right)}.$$

- A rede neural codificadora-decodificadora, contendo dois módulos LSTM, é uma máquina de tradução estatística, pois busca maximizar a probabilidade de cada frase desejada para a saída, associada a cada frase de entrada apresentada durante o treinamento.
- Seja $\left\{w_1^{(q)}, w_2^{(q)}, \cdots, w_V^{(q)}\right\}$ a sequência de palavras (símbolos) de entrada para o *q*-ésimo par de frases de treinamento, lembrando que o número *V* de palavras da frase de entrada é variável (pode ser diferente para cada *q*).
- Seja $\left\{u_{1,j_1}^{(q)}, u_{2,j_2}^{(q)}, \cdots, u_{T,j_T}^{(q)}\right\}$ a sequência de palavras (símbolos) de saída para o *q*-ésimo par de frases de treinamento, lembrando que o número *T* de palavras da frase de saída é variável (pode ser diferente para cada *q*).
- A probabilidade de cada palavra desejada para a saída é, então, dada na forma:

$$p_{\langle i \rangle, j_i}^{(q)} = p\left(u_{i,j_i}^{(q)} \middle| w_1^{(q)}, w_2^{(q)}, \cdots, w_V^{(q)}, u_{i-1}^{(q)}, u_{i-2}^{(q)}, u_1^{(q)}\right),$$

sendo j_i o índice da *i*-ésima palavra no dicionário de todas as palavras possíveis.

- E a probabilidade conjunta da frase de saída assume então a forma:

$$\prod_{i=1}^{T} p_{\langle i\rangle, j_i}^{(q)} = \prod_{i=1}^{T} p\left(u_{i,j_i}^{(q)} \middle| w_1^{(q)}, w_2^{(q)}, \cdots, w_V^{(q)}, u_{i-1}^{(q)}, u_{i-2}^{(q)}, u_1^{(q)}\right).$$

- Aplicando o logaritmo à expressão acima e somando para todo q, o ajuste dos pesos da rede neural codificadora-decodificadora, contendo dois módulos LSTM, deve maximizar o seguinte critério de desempenho:

$$\sum_{q=1}^{N}\sum_{i=1}^{T} \log\left(p_{\langle i\rangle, j_i}^{(q)}\right) = \sum_{q=1}^{N}\sum_{i=1}^{T} \log\left[p\left(u_{i,j_i}^{(q)} \middle| w_1^{(q)}, w_2^{(q)}, \cdots, w_V^{(q)}, u_{i-1}^{(q)}, u_{i-2}^{(q)}, u_1^{(q)}\right)\right],$$

lembrando que T e V variam com q.

4.2 O papel semântico do codificador

- Da mesma forma como foi verificado no caso do word2vec, o codificador da máquina tradutora de frases tem um papel semântico bem estabelecido. A figura a seguir apresenta projeções 2D do vetor de saída do codificador para algumas frases presentes na entrada, após o treinamento.

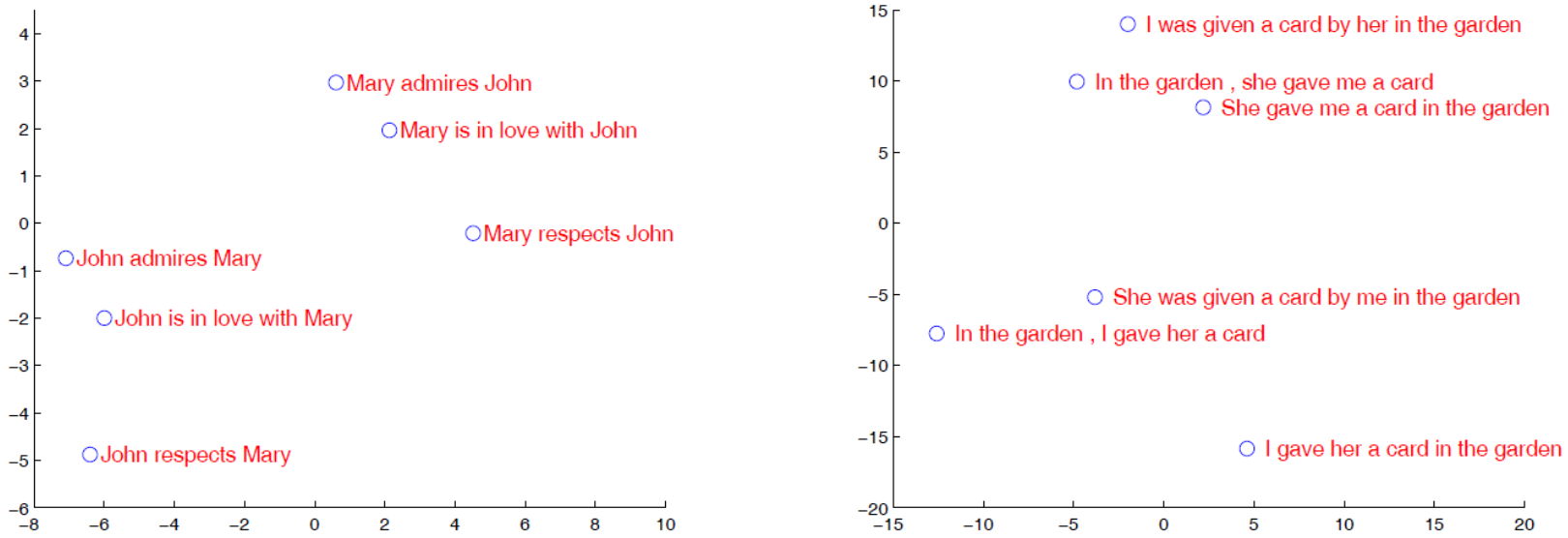
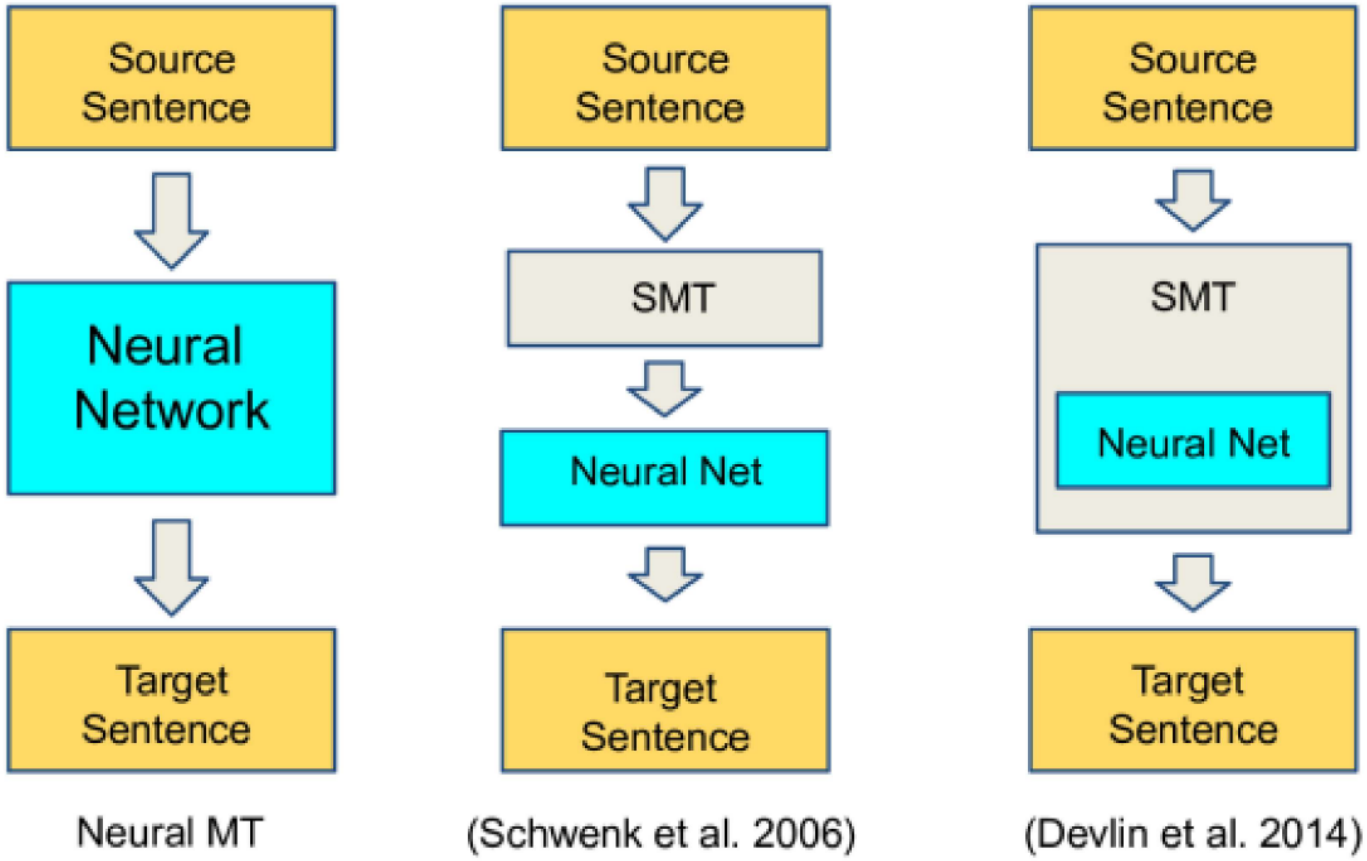


Figura 7 – Comportamento do vetor de sumarização (saída do codificador) para algumas frases de entrada (figura extraída de SUTSKEVER et al. (2014)).

- Aspectos sintáticos da linguagem, obviamente, também podem ser contemplados neste vetor de sumarização.
- Em virtude do tratamento sequencial da frase e seu mapeamento num mesmo espaço de codificação densa, é possível comparar frases de tamanho e estrutura distintos. Trata-se de um resultado muito expressivo.



- Uma rede neural passou a cuidar sozinha de toda a máquina de tradução, após aparecer como participante do processo em metodologias propostas anteriormente e que também chegaram a ser estado-da-arte.

4.3 O índice BLEU para avaliação da tradução realizada

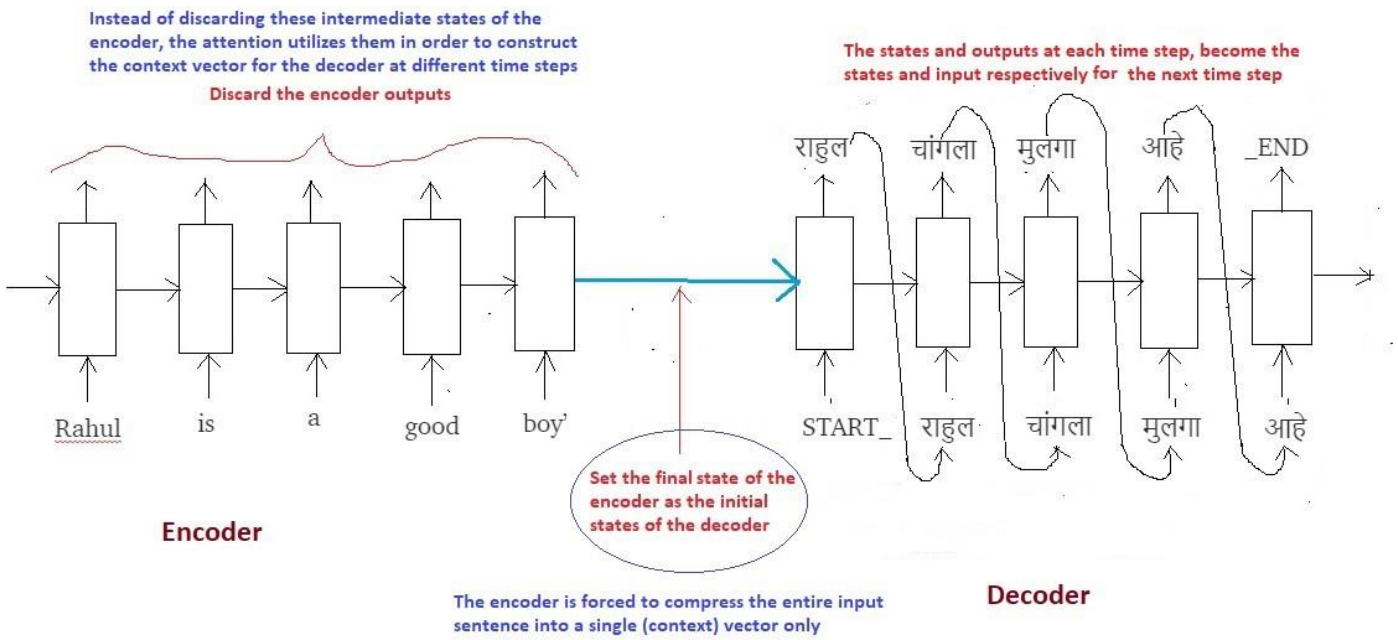
- BLEU (*BiLingual Evaluation Understudy*) (PAPINENI et al., 2002) é um índice que avalia a qualidade da tradução realizada por uma máquina, de uma linguagem natural para outra. O índice varia no intervalo [0,1] e é tão maior quanto mais próxima estiver a tradução feita pela máquina de uma tradução realizada por um especialista humano.
- O índice é uma composição das notas atribuídas a segmentos traduzidos, geralmente sentenças, pela comparação direta com um conjunto de traduções de referência para cada segmento. Todo o conjunto de segmentos do *corpus* deve ser considerado para se chegar à qualidade do resultado de tradução. Repare que inteligibilidade e correção gramatical não são diretamente ponderados.
- Cabe salientar que mesmo traduções feitas por especialistas humanos, distintos daqueles que geraram as sentenças de referência, dificilmente obtêm avaliação máxima pelo BLEU, pois muitas traduções possíveis não são consideradas.

5 Modelos de aprendizado com mecanismos de atenção

- Vamos abordar brevemente aqui uma das ideias de maior potencial de aplicação em *deep learning*: aprendizado com mecanismos de atenção (*attention learning*) (BAHDANAU et al., 2015). Dentre aplicações de destaque, cabe mencionar tradução de texto e rotulação de imagens.
- Os modelos denominados Seq2Seq, tendo a máquina tradutora de frases da seção 4 como um exemplo típico, foram os primeiros a se beneficiarem da incorporação de mecanismos de atenção. O ganho de desempenho foi tão expressivo que os tradutores estado-da-arte hoje operam apenas com eles (buscando escalabilidade, além de alto desempenho) ou tendo esses mecanismos como módulos principais.
- Será mantida aqui a aplicação envolvendo a síntese de uma máquina tradutora de texto, sendo o material desta seção predominantemente fundamentado em:

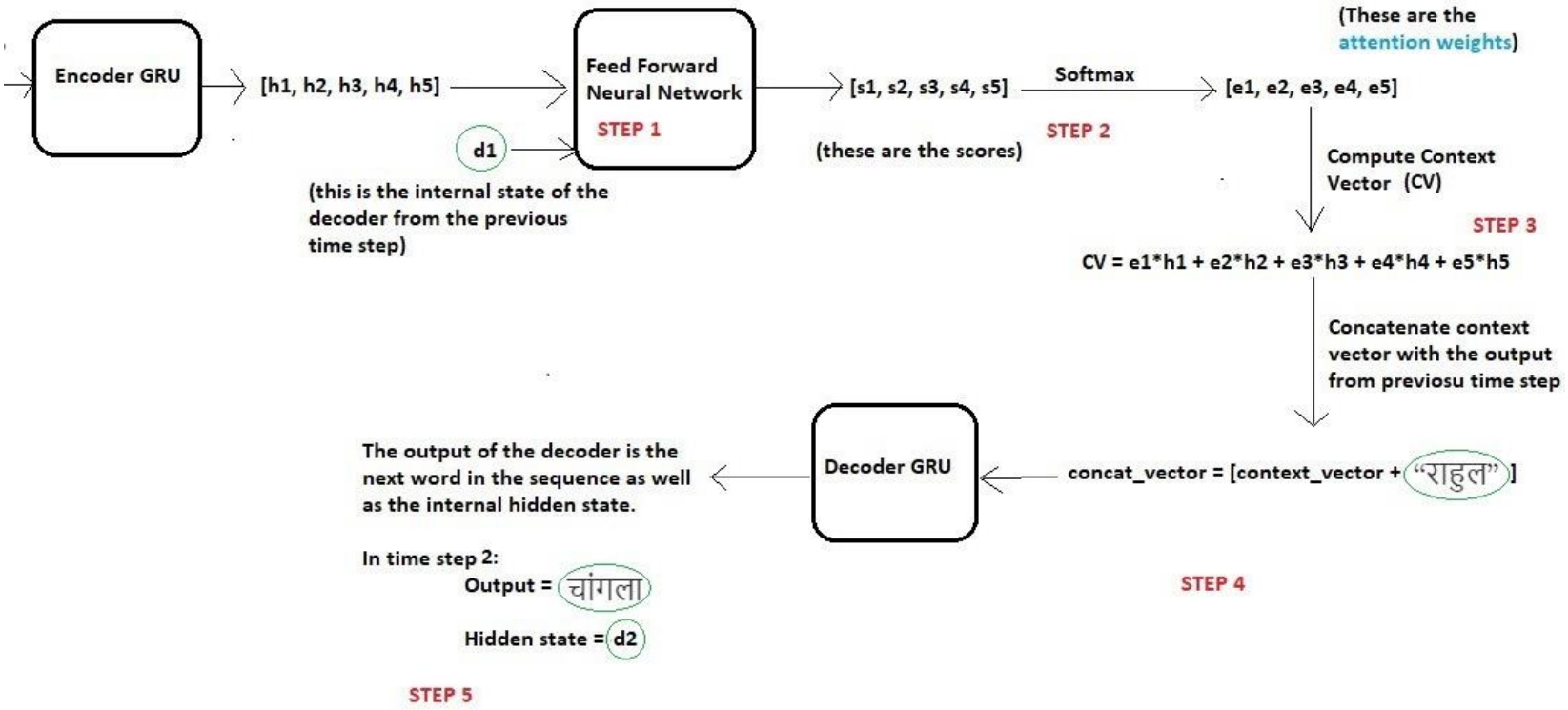
https://towardsdatascience.com/intuitive-understanding-of-attention-mechanism-in-deep-learning-6c9482aecf4f

- A arquitetura de modelos Seq2Seq, como visto na seção 4, é composta de uma estrutura codificador-decodificador. O codificador sumariza a informação de entrada (que pode ser de tamanho variável) no estado interno (que é de tamanho fixo) do bloco LSTM, também denominado de vetor de contexto. Para o sucesso da decodificação, é esperado que este vetor de contexto represente um bom sumário da sequência de entrada inteira. O decodificador, então, é inicializado com este vetor de contexto para dar partida à síntese da sequência de saída.
- Uma limitação em potencial desta proposta centrada num único vetor de contexto de tamanho fixo é que a tarefa de codificação/decodificação tende a sofrer degradação de desempenho com um aumento de tamanho da sequência de entrada.
- Os mecanismos de atenção foram justamente propostos para lidar com esta limitação. A figura a seguir ilustra a informação que é descartada quando não se adota um mecanismo de atenção.



- Todos os estados intermediários do codificador são descartados, sendo utilizado apenas o seu estado final para inicializar o decodificador. Na prática, esta estratégia se mostrou eficiente para sequências curtas e uma limitação para sequências mais longas.
- A ideia central dos mecanismos de atenção é utilizar ponderações de todos esses estados intermediários na construção do vetor de contexto.

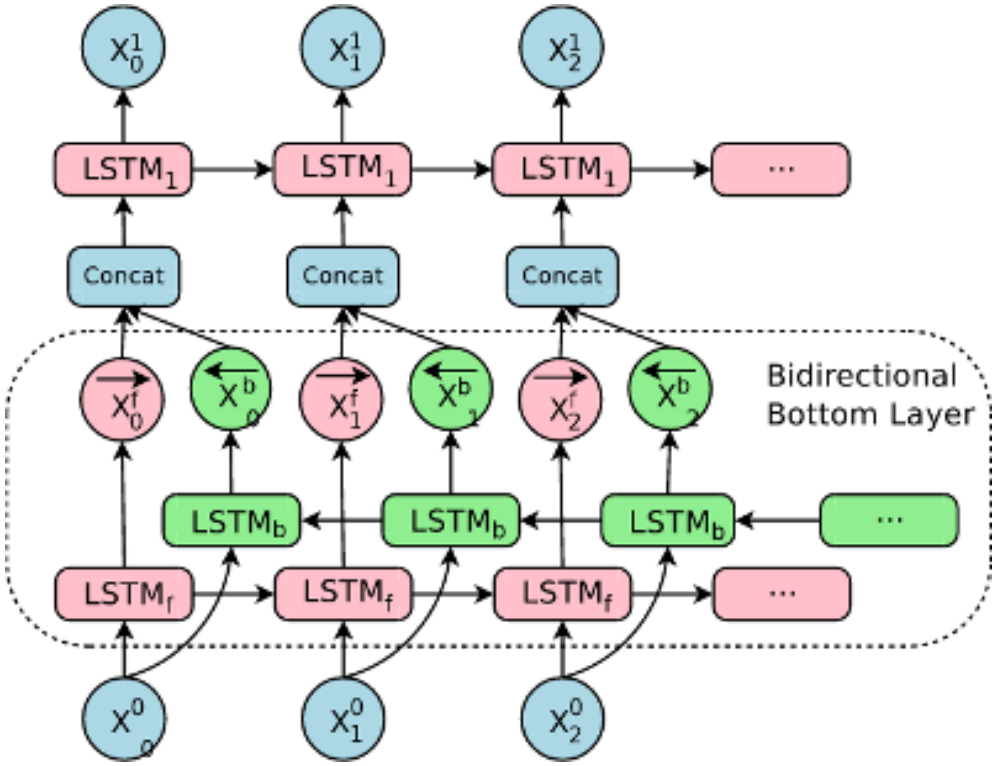
- Repare que esses estados intermediários são vetores de mesma dimensão e que armazenam informação local acerca da sequência de entrada.
- Evidentemente, não é uma tarefa elementar definir ponderações adequadas a cada instante de operação, ou seja, definir em que informações locais a atenção deve ser focada a cada instante.
- Um mecanismo de atenção competente é indiretamente obtido sempre que o comportamento de entrada-saída da máquina de aprendizado for adequadamente capturado, após o ajuste de pesos por retropropagação do erro. Em outras palavras, chega-se a um bom mecanismo de atenção ao minimizar o erro na saída da máquina de aprendizado.
- Repare que a função softmax é utilizada para a definição dos pesos de ponderação associados ao mecanismo de atenção, conforme esquematizado na figura da próxima página, onde o codificador recebe a sequência de 5 palavras “Rahul is a good boy”.



- Além de um ganho de desempenho, a introdução de mecanismos de atenção amplia a interpretabilidade dos modelos Seq2Seq, no sentido de explicar o que foi relevante, na entrada, para a definição da saída no instante corrente.
- Para mais detalhes: https://www.youtube.com/watch?v=SysgYptB198 e https://www.youtube.com/watch?v=quoGRI-1l0A

6 Codificador bidirecional

- O processamento de qualquer sequência (para efeito de tradução, por exemplo) pode depender de eventos passados e também de eventos futuros da sequência. Uma adaptação simples aos modelos Seq2Seq apresentados até aqui é a adoção de um codificador bidirecional, como ilustrado na figura a seguir.



7 Um pouco de história em PLN

- As metodologias estado-da-arte para PLN, ao longo das últimas décadas, podem ser sintetizadas na seguinte sequência:

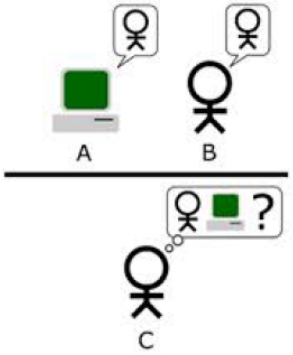
  - mid-1970s: **HMMs** for speech recognition ⇒ probabilistic models
  - early 2000s: **conditional random fields** for part-of-speech tagging ⇒ structured prediction
  - early 2000s: **Latent Dirichlet Allocation** for modeling text documents ⇒ topic modeling
  - mid 2010s: **sequence-to-sequence models** for machine translation ⇒ neural networks with memory/state

Slide extraído de Liang, P. "Natural Language Understanding: Foundations and State-of-the-Art", ICML Tutorial, 2015.

8 O jogo da imitação

The Imitation Game (1950)

"Can machines think?"



Q: Please write me a sonnet on the subject of the Forth Bridge.
A: Count me out on this one. I never could write poetry.
Q: Add 34957 to 70764.
A: (Pause about 30 seconds and then give as answer) 105621.

- **Behavioral** test
- ...of **intelligence**, not just natural language understanding

Slide extraído de Liang, P. "Natural Language Understanding: Foundations and State-of-the-Art", ICML Tutorial, 2015.

9 Detecção de sentimento

Sentiment Detection and Bag-of-Words Models

- Sentiment is that sentiment is "easy"
- Detection accuracy for longer documents ~90%
- Lots of easy cases (… horrible… or … awesome …)

- For dataset of single sentence movie reviews (Pang and Lee, 2005) accuracy never reached above 80% for >7 years

- Harder cases require actual understanding of negation and its scope and other semantic effects

Data: Movie Reviews

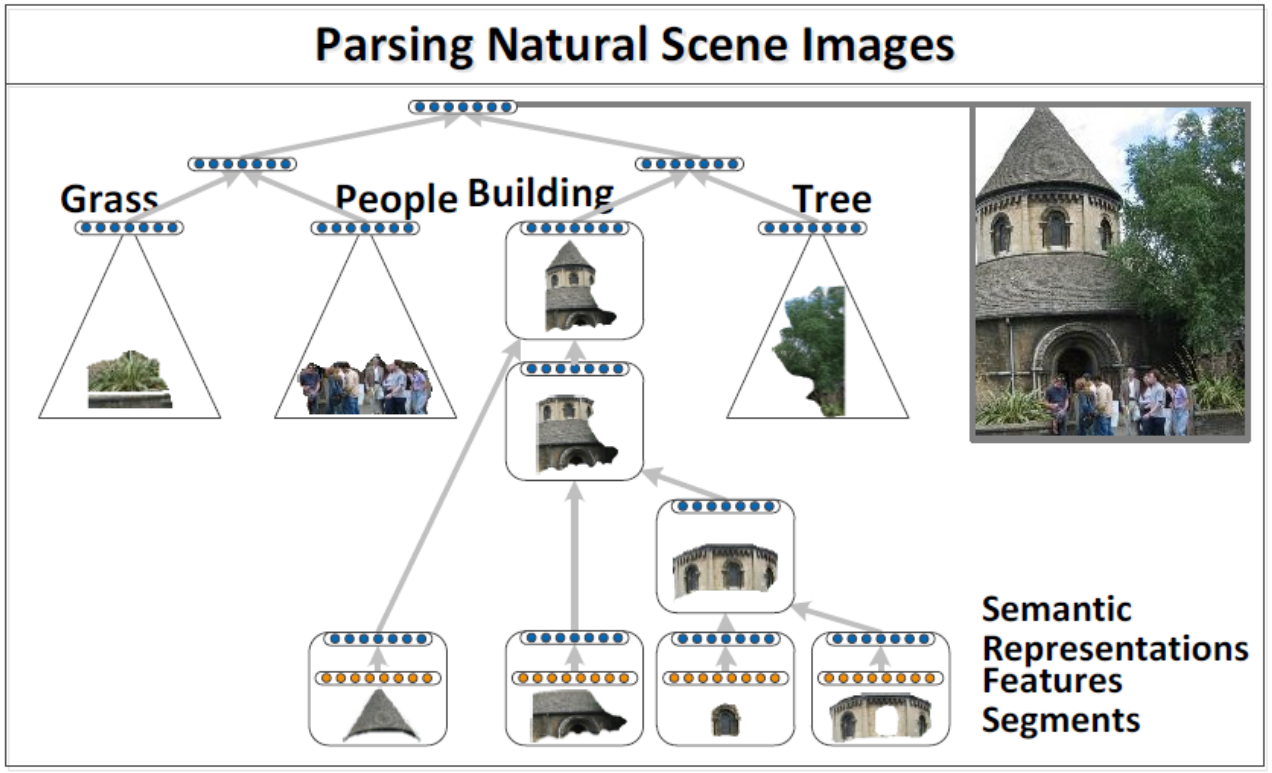
Stealing Harvard doesn't care about cleverness, wit or any other kind of intelligent humor.

There are slow and repetitive parts but it has just enough spice to keep it interesting.

10 Parsing: linguagens naturais e imagens

Algorithm for Parsing Images

Same Recursive Neural Network as for natural language parsing!
(Socher et al. ICML 2011)



11 Referências bibliográficas

BAHDANAU, D.; CHO, K.; BENGIO, Y. "Neural Machine Translation by Jointly Learning to Align and Translate", 3rd International Conference on Learning Representations (ICLR), 2015.

BARONI, M.; JOULIN, A.; JABRI, A.; KRUSZEWSKI, G.; LAZARIDOU, A.; SIMONIC, K.; MIKOLOV, T. "CommAI: Evaluating the first steps towards a useful general AI", arXiv:1701.08954, 2017.

BENGIO, Y.; DUCHARME, R.; VINCENT, P. "A neural probabilistic language model", Journal of Machine Learning Research, vol. 3, pp. 1137-1155, 2003.

CHO, K.; VAN MERRIENBOER, B.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724-1734, 2014 (também disponível em [https://arxiv.org/abs/1406.1078]).

DEERWESTER, S.; DUMAIS, S.T.; FURNAS, G.W.; LANDAUER, T.K.; HARSHMAN, R. "Indexing by latent semantic analysis", Journal of the Association for Information Science and Technology, vol. 41, no. 6, pp. 391-407, 1990.

IRSOY, O.; CARDIE, C. "Deep recursive neural networks for compositionality in language", Neural Information Processing Systems (NIPS), 2014.

LANDAUER, T.K.; FOLTZ, P.W.; LAHAM, D. "An introduction to latent semantic analysis", Discourse Processes, vol. 25, no. 2-3, pp. 259–284, 1998.

LECUN, Y.; BENGIO, Y.; HINTON, G. "Deep learning", Nature, vol. 521, pp. 436-444, 2015.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. "Efficient Estimation of Word Representations in Vector Space", In ICLR Workshop Papers, 2013a (também disponível em [https://arxiv.org/abs/1301.3781]).

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G.; DEAN, J. "Distributed representations of words and phrases and their compositionality", Neural Information Processing Systems (NIPS), pp. 3111-3119, 2013b.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.J. "BLEU: a method for automatic evaluation of machine translation", Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 311-318, 2002.

PAULUS, R.; XIONG, C.; SOCHER, R. "A Deep Reinforced Model for Abstractive Summarization", arXiv:1705.04304, 2017.

SOCHER, R.; MANNING, C.D. "Deep learning for NLP (without magic)", Tutorial at The 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, [https://nlp.stanford.edu/courses/NAACL2013/], 2013.

SUTSKEVER, I.; VINYALS, O.; LE, Q.V. "Sequence to Sequence Learning with Neural Networks", Advances in Neural Information Processing Systems 27 (NIPS), 2014 (também disponível em [https://arxiv.org/abs/1409.3215]).