CÂMARA DOS DEPUTADOS
CENTRO DE FORMAÇÃO, TREINAMENTO E APERFEIÇOAMENTO
PROGRAMA DE PÓS-GRADUAÇÃO
MESTRADO PROFISSIONAL EM PODER LEGISLATIVO

Eduardo Antônio Mello Freitas

A INTELIGÊNCIA ARTIFICIAL NA TRADUÇÃO AUTOMÁTICA DE DOCUMENTOS LEGISLATIVOS:

Uma aplicação na pesquisa interparlamentar

Brasília
2019

EDUARDO ANTÔNIO MELLO FREITAS

A INTELIGÊNCIA ARTIFICIAL NA TRADUÇÃO AUTOMÁTICA DE DOCUMENTOS LEGISLATIVOS:

Uma aplicação na pesquisa interparlamentar

Relatório Técnico-Científico apresentado ao Programa da Pós-graduação do Centro de Formação, Treinamento e Aperfeiçoamento da Câmara dos Deputados como parte dos requisitos de conclusão do Mestrado Profissional em Poder Legislativo.

Orientador: Prof. Dr. Fabiano Peruzzo Schwartz

Área de Concentração: Poder Legislativo

Linha de Pesquisa: Política Institucional do Poder Legislativo.

Brasília
2019

Autorização

Autorizo a divulgação do texto completo no sítio da Câmara dos Deputados e a reprodução total ou parcial, exclusivamente, para fins acadêmicos e científicos.

Assinatura: _____

Data: ___/___/___

Freitas, Eduardo Antônio Mello.
A inteligência artificial na tradução automática de documentos legislativos [manuscrito] : uma aplicação na pesquisa interparlamentar / Eduardo Antônio Mello Freitas. -- 2019.
132 f.

Orientador: Fabiano Peruzzo Schwartz.
Impresso por computador.
"Relatório Técnico-Científico apresentado ao Programa da Pós-graduação do Centro de Formação, Treinamento e Aperfeiçoamento da Câmara dos Deputados como parte dos requisitos de conclusão do Mestrado Profissional em Poder Legislativo."

1. Inteligência artificial. 2. Tradução automática. 3. Poder legislativo, pesquisa. 4. Poder legislativo, cooperação internacional. I. Título.

CDU 004.8:342.52

Bibliotecária: Mariangela Barbosa Lopes - CRB1: 1731

FOLHA DE APROVAÇÃO

Título: A INTELIGÊNCIA ARTIFICIAL NA TRADUÇÃO AUTOMÁTICA DE DOCUMENTOS LEGISLATIVOS: UMA APLICAÇÃO NA PESQUISA INTERPARLAMENTAR

Autor (a): Eduardo Antônio Mello Freitas

Área de concentração: Poder Legislativo

Linha de pesquisa: Política Institucional do Poder Legislativo

Trabalho de conclusão de curso submetido à Comissão Examinadora designada pela Coordenação do Programa de Pós-graduação do Centro de Formação, Aperfeiçoamento e Treinamento da Câmara dos Deputados como requisito parcial para obtenção do título de **Mestre** em Poder Legislativo.

Trabalho aprovado em 4 de julho de 2019.

Prof. Drª. Prof. Dr. Fabiano Peruzzo Schwartz
Presidente da Banca - Câmara dos Deputados

Prof. Dr. Prof. Dr. João Luiz Pereira Marciano
Membro - Câmara dos Deputados

Prof. Dr. Prof. Dr. Alexandre Ricardo Soares Romariz
Membro - UnB

Dedico este trabalho a minha esposa Elayne e a meus filhos Yasmin e Eduardo Ferreira Freitas, os quais me acompanharam nesta caminhada.

... prefiro falar cinco palavras com entendimento que alcancem os demais,
do que dez mil em linguagem desconhecida.

... prefiero hablar cinco palabras con entendimiento que alcancen a los demás,
que diez mil palabras en lenguaje desconocido.

... I prefer to speak five words with understanding that reach others,
than ten thousand words in unknown language.

θέλω πέντε λογούς διά του νοός μου λαλήσαι,
ίνα και άλλους κατηχήσω, η μυρίους λόγους εν γλώσση.

Apóstolo Paulo

Agradeço a meu orientador,
Prof. Dr. Fabiano Peruzzo Schwartz,
aos demais membros da banca,
e à equipe da DITEC pelo apoio dado a este projeto.

RESUMO

O desenvolvimento da pesquisa interparlamentar é uma forma de fortalecer a democracia por meio da cooperação entre os parlamentos. O grande volume de documentos disponibilizados apenas nas línguas oficiais é um obstáculo à pesquisa interparlamentar e à abertura mais ampla de dados. O presente trabalho tem como principal objetivo analisar a viabilidade do uso da tradução automática com vistas a facilitar a busca de conteúdo de documentos legislativos e a tornar a pesquisa legislativa menos dependente da língua. O método consiste no desenvolvimento de um tradutor automático com o uso de inteligência artificial cujo treinamento se dá a partir de documentos previamente tratados pertinentes ao Poder Legislativo; em extração de dados de sites e documentos legislativos; em pesquisa bibliográfica ampla com a finalidade de selecionar as técnicas e modelos de redes neurais artificiais mais utilizados na tradução automática; em testes de laboratório acompanhados de ajustes de parâmetros e aferição de desempenho; na apresentação em reunião perante representantes de parlamentos membros e grupos parlamentares da União Interparlamentar. O relatório introduz conceitos básicos de redes neurais relacionados aos principais modelos utilizados no desenvolvimento de tradutores automáticos e explica o preparo dos documentos. Das 8 milhões de sentenças extraídas de documentos legislativos da União Europeia, pouco mais de 5 milhões foram utilizadas nos treinamentos. O melhor resultado de treinamento alcançou 92% de acurácia na validação cruzada, o que motivou o uso da tradução automática, tanto no Brasil como em outros países, com a finalidade de disponibilizar dados e documentos legislativos em outras línguas. Conclui-se, portanto, que a tradução automática especializada cumpre com o propósito de comunicar a essência de textos legislativos e de criar novas oportunidades de pesquisa e troca de experiência entre os parlamentos e a sociedade civil mundial, respeitada a autodeterminação dos povos.

Palavras-chave: Tradução Automática Especializada. Cooperação Interparlamentar. Poder Legislativo. Inteligência Artificial. Pesquisa Interparlamentar.

ABSTRACT

The development of inter-parliamentary research is a way to strengthen democracy through cooperation between parliaments. The large volume of documents available only in the official languages is an obstacle to inter-parliamentary research and the broader opening of data. The main goal of this work is to analyze the feasibility of using machine translation to facilitate the search of content in legislative documents and to make legislative research less dependent on language. The method consists in: development of an neural machine translator with the use of artificial intelligence whose training uses documents previously prepared and pertinent to the Legislative Power; extraction of data from websites and legislative documents; extensive bibliographical research to select techniques and models of artificial neural network most appropriate for neural machine translation; parameter adjustments and performance measurement in laboratory tests; a presentation in the first meeting of the Inter-parliamentary Open Data Cloud. The report introduces basic concepts of neural networks related to the main models used in the development of automatic translators and explains the preparation of the documents. From 8 million extracted sentences from Eropean Union legislative documents, just over 5 million were used in training. The best training result achieved 92% accuracy in cross validation, which motivated the use of automatic translation, both Brazil and other countries, in order to provide data and legislative documents in other languages. It is therefore concluded that specialized automatic translation is intended to communicate the essence of legislative texts and to create new opportunities for research and exchange of experience between parliaments and world civil society, respecting the self-determination of peoples.

Keywords: Specialized Neural Machine Translation. Inter-parliamentary Cooperation. Legislative Branch. Artificial intelligence. Inter-parliamentary Research.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AI	*Artificial Intelligence*
ANN	*Artificial Neural Network*
BLEU	*Bilingual Evaluation Understudy*
BPE	*Byte Pair Encoding*
CdT	*Centre de Traduction des Organes de L'union Européenne*
DGT	*Directorate General for Translation*
DNN	*Deep Neural Network*
EC	*European Commission*
EFTA	*European Free Trade Association*
ELRC	*European Language Resource Coordination*
IBPT	Instituto Brasileiro de Planejamento e Tributação
IPU	*Inter-Parliamentary Union*
GLU	*Gated Linear Unit*
GRU	*Gated Recurrent Unit*
LSTM	*Long Short Term Memory*
MT	*Machine Translation*
NLP	*Natural Language Processing*
NMT	*Neural Machine Translation*
RNA	Rede Neural Artificial
RNN	*Recurrent Neural Network*
SGD	*Stochastic Gradient Descend*
SL	*Source Language*
SMT	*Statistical Machine Translation*
SRU	*Statistical Recurrent Unit*
TL	*Target Language*
TM	*Translation Memory*
UIP	União Interparlamentar

SUMÁRIO

1. INTRODUÇÃO

O Brasil é membro da União Interparlamentar (UIP), uma organização mundial, com escritório em Genebra, que funciona como fórum para o diálogo, cooperação e ação entre os parlamentos em prol da democracia (IPU, 2016). Atualmente a UIP conta com 178 parlamentos membros, 12 membros associados e trabalha em conjunto com as Nações Unidas dentre as organizações parceiras. Desde junho de 1955 foi regulamentada a forma de participação do Brasil com representação do Senado Federal (BRASIL, 1955a) e da Câmara dos Deputados (BRASIL, 1955b).

A fim de auferir os benefícios da pesquisa parlamentar de forma ampla e aberta, a UIP decidiu pelo desenvolvimento de uma plataforma aberta com integração de dados a fim de promover a pesquisa interparlamentar a documentos legislativos em mais de um idioma (IPU, 2017). Em 2018, a Câmara dos Deputados tornou-se responsável pela coordenação do projeto da Nuvem Interparlamentar de Dados Abertos.

O projeto tem por objetivo agregar os dados abertos publicados pelos parlamentos no mundo em uma nuvem de dados com a aplicação da tradução automática (IPU, 2019). Por facilitar a pesquisa interparlamentar e o estudo comparado, a tradução automática se apresenta como importante ferramenta para o fortalecimento dos parlamentos como instituição por meio da cooperação entre eles.

Porém o benefício não é apenas para os parlamentos, mas também para os cidadãos do mundo. O cidadão também poderá desfrutar de maior transparência, acesso e responsabilidade na prestação de contas (IPU, 2017). A tradução deve começar pelo Inglês, uma das línguas mais faladas e ensinadas no mundo. Contudo, a língua inglesa pode não ser tão acessível ao cidadão. A participação da sociedade civil organizada se fará necessária.

A língua, por certo, é um instrumento poderoso de comunicação, mas a existência de várias línguas diferentes acaba por dificultar a comunicação no mundo. Conforme a organização sem fins lucrativos SIL Internacional, existem mais de sete mil línguas faladas no mundo (SIMONS; FENNING, 2018), porém a maioria dessas línguas não foi adotada oficialmente. Contudo, quando um país adota oficialmente uma língua, esta passa a ser reconhecida como um idioma daquele país (VICHESSI, 2018). A partir de então, passa-se a exigir que os documentos oficiais sejam escritos nesses idiomas.

A entrada de um novo idioma no contexto oficial também pode ocorrer por conta da assinatura de acordos ou tratados internacionais. Tais documentos deverão estar escritos nas línguas correspondentes aos idiomas dos países envolvidos. O maior esforço é que o conteúdo dos referidos documentos não dependa do idioma em que foram escritos. A própria concepção de um acordo parte de premissa de que houve entendimento entre as partes. Tal entendimento deverá ficar claro para governantes e governados.

Ademais, a experiência de livre comércio entre países, ou nações, pode ir além das relações de fronteiras, trocas de mercadorias e acordos de tributação. Parcerias podem ser criadas entre as empresas dos países envolvidos, além do compartilhamento de outros elementos, como o modo de fazer e ver as coisas, os hábitos e as normas. Um exemplo que se destaca no mundo quanto ao convívio entre povos que fazem uso de diferentes línguas é o caso da União Europeia.

1.1 MULTILINGUISMO E O PARLAMENTO EUROPEU

A União Europeia, por ter um parlamento supranacional, é um excelente exemplo dos desafios a serem enfrentados, uma vez que estão reunidos países com diferentes idiomas, histórias e culturas. A abrangência dessa união é tal que pessoas, bens, serviços e capital chegam a circular sem fronteiras, praticamente como se fossem um único país (UNIÃO EUROPEIA, 2018b). Para tal convivência existem regras, as quais devem harmonizar as relações com base no respeito à cultura de cada povo e no bem comum. Por esse motivo, são disponibilizados, em pelo menos 24 idiomas, documentos legais e jurídicos envolvendo os países membros da União Europeia. O acesso a documentos do Parlamento Europeu pode ser feito pelo portal internet denominado EUR-Lex (UNIÃO EUROPEIA, 2018a). Nele podem ser encontrados documentos do direito da União Europeia tais como: tratados, atos jurídicos, textos consolidados, acordos internacionais, documentos da Associação Europeia de Livre Comércio (EFTA) e jurisprudências.

A depender da data e do assunto, os documentos podem ser encontrados nos vinte e quatros idiomas oficiais da União Europeia. Contudo, além desses idiomas, é possível ainda encontrar documentos em idiomas de países com os quais a União Europeia tenha relações mais estreitas, ou pode ainda incluir línguas utilizadas por grupos específicos, como é o caso de um movimento de imigração. No EUR-Lex, ainda, é possível ver alguns documentos

alinhados, ou em paralelo, em até três idiomas distintos de forma alinhada.

A questão da língua é de tal relevância que a Comissão Europeia, corpo executivo da União Europeia, criou uma diretoria-geral específica para tratar de tradução, a Diretoria-Geral de Tradução (DGT) (UNIÃO EUROPEIA, 2016). A DGT investe em tradutores humanos e tecnologia, em que a tradução automática é vista como um recurso de apoio à tradução. O provimento de tais recursos é alvo de grande investimento financeiro realizado pela União Europeia. Conforme constou no jornal Slators, especializado no mercado de tecnologias de tradução e de linguagem, o Centro de Tradução dos Organismos da União Europeia (CdT) traduziu cerca de três quartos de milhão de páginas em 2016. O orçamento do CdT para 2018 foi estimado em 47,7 milhões de EUR (ESTOPACE, 2017). Há também a Coordenação de Recursos Linguísticos Europeus (*European Language Resource Coordination* – ERLC) que compartilha recursos de apoio à tradução. A finalidade desses recursos é facilitar o trabalho de tradução com o aproveitamento de traduções anteriores (UNIÃO EUROPEIA, 2017).

1.2 A PESQUISA INTERPARLAMENTAR E A TRADUÇÃO

A experiência da União Europeia mostrou a importância da tradução como forma de reduzir as barreiras advindas do desconhecimento de uma determinada língua. Outro aspecto a considerar é o incentivo à cooperação entre pessoas e instituições. Trata-se de um trabalho que deve contar com a união de esforços e cooperação entre os países.

Ora, a disponibilidade de dados para a pesquisa de dados legislativos, no que diz respeito às normas escritas, é facilitada em parte pelo requisito de publicidade. Para que uma norma escrita seja obedecida, ou cumprida, ela precisa ser conhecida publicamente. Porém, em se tratando de instituições públicas sob os princípios democráticos, a abertura não deve se limitar à publicidade. A pesquisa legislativa pode se tornar uma via de mão dupla se a abertura não for apenas de dados, mas incluir também a própria instituição (FREITAS, 2017). A pesquisa interparlamentar deve ser vista não apenas como uma forma de consumir dados, mas como um compromisso de compartilhar conhecimento por parte dos parlamentos. Desse modo, a pesquisa interparlamentar pode se tornar um espaço aberto para a socialização e a externalização do conhecimento (WANG; WANG, 2012).

Conquanto a Lei seja o resultado final do processo legislativo, ainda assim ela é um

documento capaz de revelar informações que vão além da norma escrita. Documentos legislativos, em geral, tratam de procedimentos, obrigações, direitos, definições, orçamentos e estruturas. Tais documentos são resultado de decisões tomadas em conjunto e, ainda que alguns elementos do processo legislativo possam ser desconhecidos, eles não deixam de ser uma forma de explicitar as preocupações ou interesses imediatos, a experiência e o modo de fazer dos legisladores. Esses elementos também estão presentes no conhecimento tácito, os quais, segundo Nonaka e Takeuchi (1997), devem ser externalizados a fim de promover a criação do conhecimento. Além do mais, lições podem ser depreendidas da história das leis sobre determinado tema. Por exemplo, se uma lei, logo após o início de sua vigência, precisou sofrer alterações ou foi revogada, isso é um alerta quanto à forma de lidar com um determinado assunto. Analisar a história de uma lei também é uma forma de aprendizado, o que pressupõe acertos e erros. O pesquisador deverá considerar a abordagem cuidadosa do tema pesquisado e buscar extrair as lições aprendidas, ainda que estas não estejam apresentadas como tais.

1.3 O PAPEL DO BRASIL

A Câmara dos Deputados foi escolhida como *hub* temático, ou especializado, responsável pelo projeto da Nuvem Interparlamentar de Dados Abertos. Os *hubs* são polos de pesquisa e de inovação que integraram o Centro de Inovação para os Parlamentos da UIP. A primeira tarefa do Brasil foi a de pesquisar alternativas de tradução dos dados. A língua inglesa foi escolhida como a primeira língua comum da pesquisa interparlamentar. Apesar da primeira etapa se reduzir a apenas uma língua adicional, países que não têm o inglês por língua oficial e que aplicam poucos recursos em tradução permaneceram com um grande desafio a ser enfrentado.

O Brasil se configura entre os países que não possuem a língua inglesa por idioma oficial e que aplica pouquíssimos recursos na tradução de documentos oficiais. Além do mais, de acordo com o Instituto Brasileiro de Planejamento e Tributação (IBPT), desde a data da promulgação da Constituição Federal de 1988 até a data de 30 de setembro de 2016, foram editadas 5.471.980 normas. No âmbito federal foram editadas 163.129 normas gerais no mesmo período (IBPT, 2016). O número de normas continua a crescer apenas na língua portuguesa, língua oficial do Brasil. Algumas poucas leis podem ser encontradas em inglês,

mas várias delas estão em versões já desatualizadas, como é o caso de versões da Constituição Federal de 1988 disponíveis em inglês na internet.

1.4 A TRADUÇÃO AUTOMÁTICA

Diante do grande volume de leis, contar apenas com a tradução humana é viável em curto prazo. Ainda que fossem adquiridos produtos de apoio à tradução humana, tais recursos seriam ineficientes diante do volume de projetos de leis e normas. O uso da tradução automática desponta como alternativa devido a seu uso cada vez mais comum nos ambientes *online*. Juntas, a internet e a tradução automática diminuem as barreiras da distância e da língua na comunicação. De acordo com Hutchins (1997), a tradução pode atender a dois tipos de necessidade:

1) obter uma tradução com garantias e de alta qualidade para fins de publicação, caso em que os recursos tecnológicos funcionam como ferramenta de apoio na tradução;

2) obter uma tradução capaz de transmitir a essência do texto original, ainda que alguns erros possam ser encontrados, caso em que os recursos tecnológicos podem responder por toda a tradução.

Hutchins (2017, p. 1) denominou as máquinas de tradução que atendem ao primeiro tipo de necessidade de "máquinas de tradução para a divulgação[1]". Às demais, deu-lhes o nome de "máquinas de tradução para a assimilação". Todavia, em se tratando de troca de mensagens sociais por e-mail, canal de bate-papo ou outro tipo de rede social, chamou-as de "máquina de tradução para comunicação". O desenvolvimento de soluções de tradução automática com o uso da inteligência artificial aponta para uma qualidade cada vez maior das ferramentas de tradução. Conquanto a qualidade dos resultados alcançados com experimentos em sistemas de tradução sugira uma qualidade próxima da média de tradutores humanos (WU, 2016), a proposta é o uso da tradução automática para fins de assimilação da essência do texto.

A proposta em prol do uso da tradução automática foi apresentada logo após a escolha do Brasil como responsável pelo projeto da Nuvem Interparlamentar de Dados

[1] Preferiu-se a tradução do termo original *dissemination* como "divulgação", por razões etimológicas, pois não se pode dizer que não há disseminação ao se levar a essência do texto. Por outro lado, a divulgação presume uma responsabilidade maior com a escrita dentro do estilo escolhido.

Abertos. O autor deste relatório esteve em Bruxelas em junho de 2018, quando foram marcados encontros com a equipe de tecnologia do Parlamento Europeu e com uma equipe envolvida com a tradução na Universidade de Frankfurt. A reunião com os desenvolvedores da Universidade de Frankfurt foi por eles cancelada. Houve ainda duas apresentações, uma sobre técnicas utilizadas na tradução direta, ainda com o modelo estatístico de aprendizagem de máquina, e outra na Semana de Inovação nas dependências do Parlamento Europeu pelo Instituto de Tecnologia de Karlshure. Na época, já haviam sido extraídos os índices de todos os documentos legais legislativos do EUR-Lex que continham os textos em Português e Inglês, o que posteriormente veio a totalizar 8 milhões de sentenças em Português, Inglês, Espanhol, Francês e Italiano.

1.5 QUANTO À ESTRUTURA DO RELATÓRIO

Por envolver conhecimentos que normalmente não fazem parte das atividades do Poder Legislativo, nem das disciplinas geralmente estudadas em Ciência Política, estão presentes no capítulo Desenvolvimento do relatório dois tópicos que introduzem conceitos e estruturas de redes neurais artificiais (2.3) e de aspectos linguísticos importantes para a tradução automática (2.4). O tópico “Ponto de Partida para a Codificação” (2.5), trata dos recursos de hardware e de software necessários ao desenvolvimento do tradutor.

Os procedimentos de desenvolvimento (2.6.1) aplicados no laboratório de tradução automática seguem logo após a descrição da metodologia adotada. Aos leitores que queiram se aprofundar mais sobre a codificação, os recursos utilizados, cálculos, parâmetros e a apresentação à União Interparlamentar, recomenda-se a leitura dos apêndices e dos anexos.

2. DESENVOLVIMENTO

A considerar os benefícios da pesquisa interparlamentar e os avanços da tradução automática, a solução mais rápida de obter traduções de documentos legislativos seria o desenvolvimento de uma ferramenta de tradução automática especializada em documentos legislativos com a utilização da Inteligência Artificial. As traduções resultantes não terão valor legal, mas serão de grande valia para o pesquisador que desconhece o idioma original. Ademais, tal solução pode ser compartilhada por outros países.

2.1 OBJETIVO GERAL

O objetivo geral é desenvolver, com recursos da inteligência artificial, uma ferramenta de tradução automática treinada a partir de documentos do contexto do Poder Legislativo, a fim de facilitar a pesquisa a documentos legislativos em mais de uma língua.

2.1.1 OBJETIVOS ESPECÍFICOS

Os objetivos específicos são os seguintes:

- Confirmar a viabilidade da tradução automática para a pesquisa interparlamentar.
- Identificar os requisitos mínimos para o funcionamento de um tradutor automático.
- Propor e desenvolver técnicas de pré-processamento de documentos legislativos a fim de aumentar a qualidade da tradução.
- Elaborar um protótipo de página Web para disponibilizar a tradução online.

2.2 ESCOPO DO RELATÓRIO

O escopo do projeto leva em consideração os documentos legislativos, os idiomas para a tradução e os tipos de formato dos documentos a serem traduzidos. Para fins de simplificação, documentos legislativos publicados com força de lei serão simplesmente

chamados de Lei. Igualmente, a toda proposição ou minuta destinada a se tornar uma Lei, este relatório irá se referir como Projeto de Lei.

2.2.1 QUANTO AO TIPO DE TREINAMENTO

Conquanto existam interessantes estudos sobre o treinamento não supervisionado para fins de tradução, eles foram considerados ainda incipientes para a tradução automática. O treinamento não supervisionado em uma língua implica tentar mapear os campos semânticos e aproximá-los por suas propriedades, sem que haja um "professor" dizendo se está certo ou errado. De certa forma, a conversão das palavras em vetores tem aspecto de aprendizagem não supervisionada. Porém, a escolha final das palavras passa pela avaliação do resultado encontrado em relação ao esperado.

Ao se optar pela tradução automática com o treinamento supervisionado, para cada sentença ou palavra no idioma de entrada é esperada uma sentença ou palavra equivalente no idioma de saída. A rede deverá procurar corrigir os pesos a fim de minimizar o erro, e a essa operação será aplicada uma taxa de aprendizagem (*learning rate*, em inglês). Assim sendo, os arquivos no idioma de entrada, ou SL (*source language*), e os arquivos no idioma saída, ou TL (*target language*), devem ter uma correspondência linha a linha de mesmo conteúdo, ainda que em idiomas diferentes.

2.2.2 QUANTO ÀS FONTES DOS DOCUMENTOS

Quanto aos documentos utilizados para a aprendizagem da rede de inteligência artificial, estes devem ser pertinentes aos temas e ao contexto do processo legislativo. Desse modo, espera-se que a restrição reduza o universo de palavras e expressões e que a linguagem se torne mais especializada, também chamada de sublinguagem (COULTHARD; CALDAS-COULTHARD, 1991; MARTINS, 2009). O treinamento da tradução automática exige um grande volume de textos em pelo menos dois idiomas. Uma vez que a maior parte dos documentos obtidos teve origem no Parlamento Europeu, foram incluídos textos paralelos contendo do legislativo brasileiro. Uma versão em inglês da Constituição da República Federativa do Brasil de 1998 (BRASIL, 2016b) publicada pela Câmara dos Deputados foi utilizada a fim de viabilizar o uso correto dos termos técnicos do Poder Legislativo.

Os tipos e formatos dos documentos do processo legislativo dependem das leis de cada país. O caminho do processo legislativo para a formulação de uma lei pode depender do assunto tratado, do tipo de projeto de lei e dos órgãos que podem propor leis. O próprio poder legislativo pode ser unicameral ou bicameral. No caso do Brasil, o poder legislativo federal brasileiro corresponde ao Congresso Nacional. Conforme o artigo 44 da Constituição da República Federativa do Brasil (BRASIL, 2016a), o Congresso Nacional é composto por duas casas legislativas: a Câmara dos Deputados e o Senado Federal. Diferenças nas tradições, nos regimentos, nas definições e nos fluxogramas internos do processo legislativo federal da Câmara dos Deputados e do Senado Federal refletem a autonomia interna naquilo que não foi previamente definido na Constituição, conforme artigo 51, inciso III e artigo 52, inciso XII (BRASIL, 2016a). Se a autonomia entre duas casas legislativas de um mesmo país resultou em particularidades no encaminhamento do processo legislativo, o mais provável é que sejam encontradas diferenças em se tratando de países distintos. Decidiu-se, portanto, pelo afastamento de detalhes do processo legislativo, o que limitou os documentos àqueles do início e fim do processo legislativo. Portanto, foram consideradas para início dos testes de tradução, as primeiras versões dos projetos de lei e as leis que foram publicadas.

2.2.3 QUANTO AOS IDIOMAS

Idioma é a língua própria de uma nação. O Português é o idioma oficial do Brasil. Diferente de países como a Suíça — que possui quatro línguas reconhecidas pelo artigo 4º da Constituição de 1999 (SUÍÇA, 2018) como idiomas oficiais (alemão, francês, italiano e romanche) — e o Canadá — com em duas línguas oficiais (inglês e francês) — o Brasil não conta com pouquíssimos recursos multilíngues. A presença de tradutores em órgãos públicos também não é muito comum no Brasil. Assim sendo, quanto ao idioma, o escopo compreende a tradução automática entre o Português e o Inglês. Porém, nada impede que sejam os mesmos procedimentos sejam aplicados com documentos em outras línguas.

Apesar de possuírem o mesmo idioma oficial, Brasil e Portugal fazem uso de variantes próprias da língua portuguesa. O Acordo Ortográfico acabou por aumentar o número de homógrafos ao retirar acentos gráficos de paroxítonas que serviam para distingui-los e manteve mais de uma forma de escrita para várias palavras. Tais diferenças são levadas em conta no treinamento.

2.2.4 QUANTO AOS FORMATOS PARA TRADUÇÃO

Os documentos legislativos devem ser traduzidos primariamente a partir de arquivos de texto sem formatação. A fim de obter mais dados de treinamento, foram aplicadas técnicas de extração textos de arquivos em formato HTML. Alguns dados foram copiados de arquivos no formato DOC (Microsoft Word), como foi o caso da Constituição da República Federativa Brasileira, a fim de enriquecer o vocabulário com as terminologias utilizadas no processo legislativo brasileiro. A princípio, não foi considerada como parte do escopo deste relatório a conversão de formatos de documentos no processo de tradução além da extração.

O uso do tradutor na modalidade de serviço inclui uma aba para a apresentação em formato HTML simples. Documentos em formato em XML também podem simples, contendo apenas as marcações, podem ser traduzidos.

2.2.5 QUANTO AO PROJETO BASE

O projeto base escolhido foi o Open Neural Machine Translation (OpenNMT), que, além de ter o código aberto, já foi utilizado com sucesso no 2º Workshop em Máquinas Neurais de Tradução 2014 (SENELLART *et al*., 2018). A versão em Python traz em sua licença a referência ao relatório elaborado pela equipe do projeto OpenNMT-py (KLEIN *et al*., 2017).

A versão OpenNMT-py foi utilizada e adaptada para as principais operações de tradução propriamente ditas. Alguns programas e conceitos foram aproveitados da versão original OpenNMT escrita na linguagem de programação LuaTorch.

O projeto OpenNMT-py também é capaz de aceitar imagem ou áudio como forma de entrada, o que poderá ser explorado na tradução em eventos.

2.3 REDES NEURAIS ARTIFICIAIS

Em um sistema de computação clássico especialista, as decisões são previamente programadas com base em arquiteturas ou modelos especializados que tomam as decisões a partir dos resultados de parâmetros, algoritmos e cálculos previamente escolhidos. A confiabilidade do sistema dependerá de sua programação e de serem mantidas as condições do projeto. A primeira geração de sistemas inteligentes contou com o apoio de modelos estatísticos. Segundo Feigenbaum (1981, p.1), "o conhecimento de um sistema especialista consiste de fatos e heurística". Os sistemas especialistas procuravam fazer uso do conhecimento a fim de desenvolver a inteligência artificial ao ponto de poder inferir as melhores escolhas a partir dos dados fornecidos. O primeiro registro de um projeto de pesquisa desenvolvido sobre inteligência artificial foi em 1955, no Dartmouth College, com o título *Artificial Intelligence* (AI), o qual foi apresentado, na Universidade de Stanford em 1956, por John McCarthy, Marvin Minsky, Nathaniel Rochester e Claude Shannon. O projeto, em questão, se preocupou com a o modo como o ser humano lida com a linguagem no pensamento (MCCARTHY *et al*., 2006), aspecto essencial para a tradução automática nos dias de hoje. Contudo, o desenvolvimento de máquinas de tradução partiu de modelos estatísticos com as chamadas máquinas estatísticas de tradução (SMT, do inglês *Statistical Machine Translation*). As SMT são utilizadas até hoje como ferramentas de apoio à tradução.

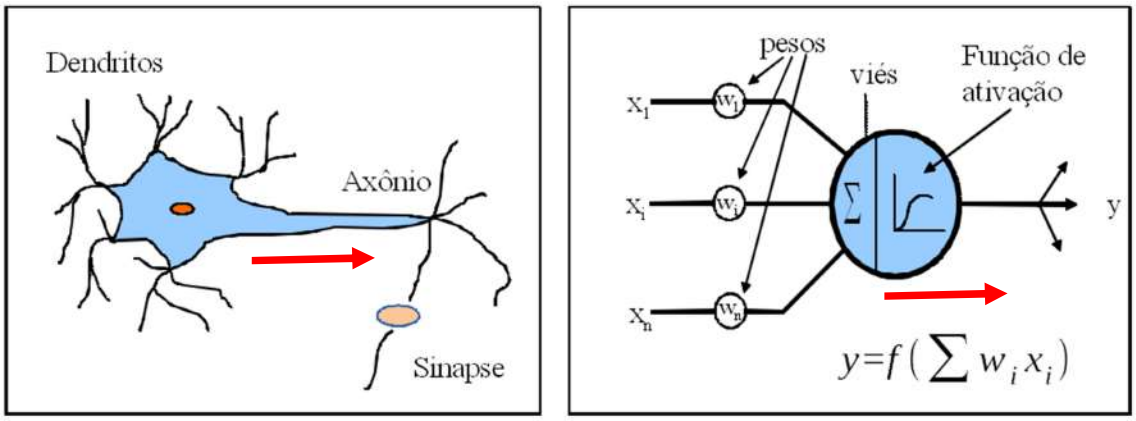
Diferentemente das SMT, as máquinas de tradução automática (NMT, do inglês *Neural Machine Translation*) aprendem o próprio modelo a ser aplicado. Logo, nas redes neurais, a "programação" é fruto do aprendizado. A concepção das redes neurais artificiais parte de uma analogia com as redes neurais biológicas (ROQUE, 2016).

2.3.1 A ANALOGIA COM AS REDES BIOLÓGICAS

As redes neurais artificiais foram inspiradas nas redes neurais biológicas com o objetivo de promover uma aprendizagem mais profunda, capaz de se adaptar, reconhecer, classificar e agrupar padrões e séries temporais (ROQUE, 2016). Trata-se de uma versão bastante simplificada de uma rede biológica.

Na Figura 1, no retângulo à esquerda, está uma representação simples de um neurônio biológico em que estão destacados: os dendritos, que são canais de entrada e transformação da informação; o axônio, que processa os pulsos de saída e os transmite com seus terminais; e uma região denominada sinapse.

Figura 1- Neurônio Biológico e Artificial



Fonte: GUDWIN, 2003 (adaptado pelo autor).

Os neurônios podem se comunicar com outras células e com outros neurônios, mas sempre em um sentido. Na comunicação entre neurônios, os sinais de saída dos neurônios, também chamados de pré-sinápticos, são provenientes dos terminais do axônio. Os sinais atravessam a sinapse, região de conexão entre os neurônios, até chegar ao dendrito do próximo neurônio (LI; JOHNSON; YEUNG, 2018).

Os sinais de entrada provenientes de outro neurônio, ou dele mesmo, são denominados pós-sinápticos. Eles são recebidos por meio dos dendritos ou entregues diretamente na região do neurônio denominada soma, a qual é responsável por agregar todas as informações recebidas (LI; JOHNSON; YEUNG, 2018).

Os sinais que os neurônios biológicos recebem e transmitem podem ser químicos, elétricos ou ambos. Eles podem ter por objetivo excitar ou inibir. Pesquisas demonstram a existência de processamento em paralelo entre neurônios, isto é, mais de um neurônio pode receber sinais de uma mesma região, processá-los e enviá-los para o mesmo neurônio de destino. Isso significa que, na falha de um neurônio, outro pode executar a mesma função, tal como em um sistema de tolerância a falhas. Além disso, existem recursos de balanceamento de sinais, o que pode ser feito por meio de ajustes no sincronismo entre os sinais de entrada e saída, na amplitude do sinal ou ainda na frequência do sinal (LONDON; HÄUSSER, 2005).

É importante insistir que, conquanto os neurônios biológicos tenham servido como inspiração para os neurônios artificiais, as funcionalidades dos neurônios biológicos não se resumem às suas analogias artificiais. Existem muitos tipos de neurônios biológicos com diferentes tipos de estruturas de dendritos e eles são capazes de processar operações complexas lineares e não lineares. Além do mais, as próprias sinapses são mais do que funções ou pesos nas entradas dos neurônios (LI; JOHNSON; YEUNG, 2018). Quando uma pessoa aprende algo novo, o cérebro faz alterações dinâmicas nas sinapses, nas atividades cerebrais de certas regiões e até mesmo nos circuitos cerebrais em larga escala (BASSETT *et al*., 2013; ROSENBLATT, 1986).

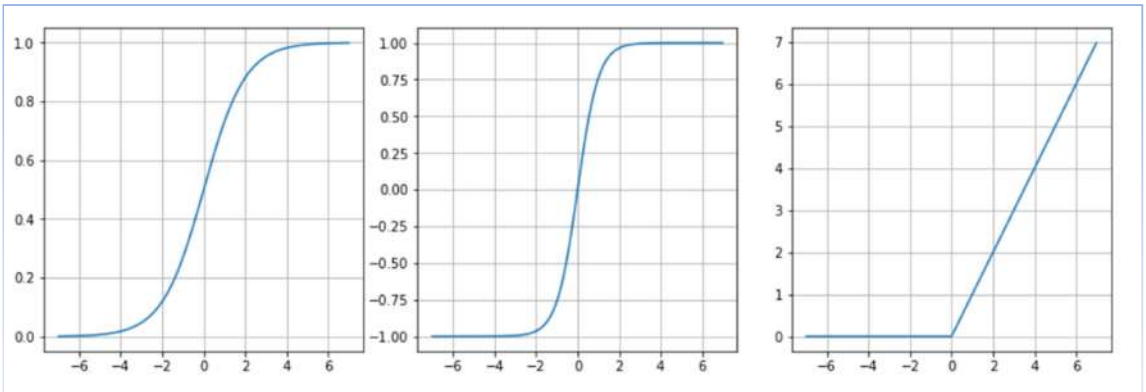
O cérebro humano chega a cerca de 100 bilhões de neurônios (VON BARTHELD *et al*., 2016) e seu consumo é de até cerca de 20 watts, segundo pesquisa publicada na revista The Economist (2013). Quando comparados os cérebros de mamíferos com computadores de alto desempenho, no quesito consumo de energia, os mamíferos estão muito a frente (STRUKOV, 2011).

2.3.2 OS NEURÔNIOS ARTIFICIAIS

De volta à Figura 1, agora no retângulo à direita, é apresentado um neurônio artificial. As entradas recebem valores de x_1 a x_n, em que n é o número de entradas. Um valor pode ser incluído para o ajuste do viés, ou nível de entrada (x_0 ou b, de *bias* em inglês). Normalmente, os produtos dos valores de entrada pelos respectivos pesos (w_1 a w_n), também chamados de pesos sinápticos, são somados ($X = x_0 + \sum_{k=1}^{n} x_k . w_k$) e é aplicada uma função que é chamada de função de ativação. O valor Y é o resultado da função de ativação (GUDWIN, 2003).

As funções de ativação, em sua maioria, não são lineares. Considerando o valor de $X = x_0 + \sum_{k=1}^{n} x_k . w_k$, então a função de ativação corresponde a $Y = f(X)$. A Figura 2 mostra os gráficos das principais funções de ativação.

Figura 2- Funções de Ativação: Logística, tanh e ReLU



Fonte: Elaboração própria, 2018.

As funções logística, tangente hiperbólica e ReLU, representadas da esquerda para a direita na figura 2, fazem uso das seguintes equações:

- Logística: $f(x) = 1 / (1 + \exp(-x))$, com saída variando entre 0 e 1.
- Tangente hiperbólica: $f(x) = 2 / (1 + \exp(-2x)) - 1$, com saída entre -1 e 1.
- ReLU: $f(x) = \max(0, x)$, com saída de 0 ao maior valor de x.

Outra importante função utilizada em redes neurais artificiais é a *Softmax*, que é utilizada para normalizar valores e facilitar a classificação ou escolha do melhor valor dentro de um conjunto de valores. Supondo que a última camada tenha n saídas, para cada valor y_i, com $i \in \{1, \dots, n\}$, a função *Softmax* é expressa na Equação 1.

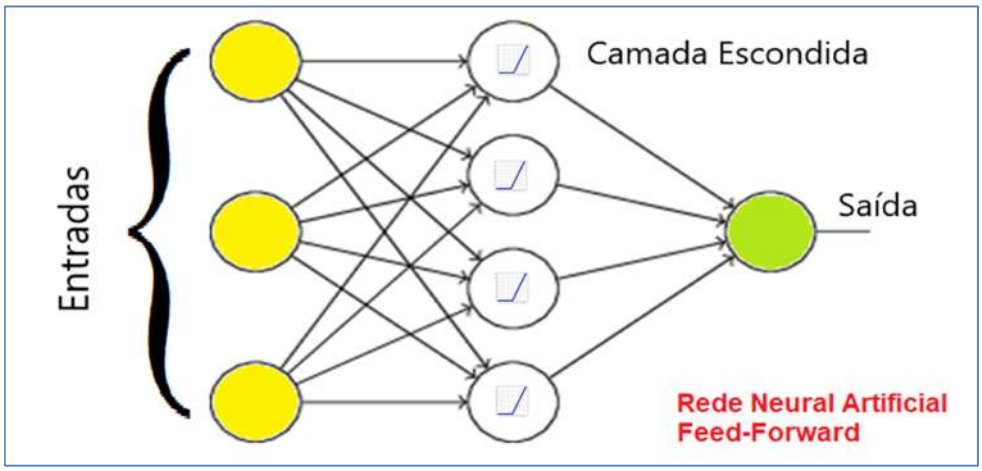
$$Softmax(y_i) = \frac{e^{y_i}}{\sum_1^n e^{y_j}} \qquad (1)$$

2.3.3 ARQUITETURA DA REDE NEURAL ARTIFICIAL

Diferentemente de métodos mais tradicionais de programação ou adequação de sistemas especializados, o treinamento supervisionado de uma rede neural artificial (RNA) não parte de um algoritmo previamente codificado como um programa de computador. Nem tão pouco de modelos de cálculo previamente preparados e adequados ao comportamento das amostras, como os usados em aprendizagem de máquina. No treinamento supervisionado de uma rede neural artificial são necessários os valores de entrada e os valores da saída esperada.

As redes neurais artificiais podem ser analisadas conforme a disposição de suas camadas e interligações, podendo contar com um ou mais neurônios artificiais. A Figura 3 apresenta como exemplo uma rede denominada de Feed-Forward.

Figura 3- Arquitetura de Redes Neurais Artificiais



Fonte: Elaboração própria, 2019.

Em uma rede Feed-Forward o processamento das entradas é feito em um único sentido e todas as saídas de uma camada se conectam às entradas da próxima camada. A camada mais interna é chamada de camada "escondida", indicado na Figura 3. Distinguir entre a aprendizagem profunda (*deep learning*) e a aprendizagem rasa (*shallow learning*) não tem a ver com a complexidade do problema, mas com a arquitetura da rede. Na Figura 3, por ter menos de duas camadas escondidas, o processamento segue apenas em um sentido, a aprendizagem não é considerada profunda. A aprendizagem está em reduzir o erro entre a saída esperada e a saída prevista. A atualização dos valores dos pesos sinápticos pode ser feita a cada cálculo de erro ou em lote, por meio de uma função de custosão atualizados são estimados e aplicados a cada entrada dos neurônios a fim de encontrar o resultado desejado com o menor erro (SCHMIDHUBER, 2015). À medida que os pesos da rede são ajustados ela, com os seus estados, se torna em um modelo.

Nas seções seguintes serão apresentados os principais modelos de redes neurais artificiais empregados e algumas peculiaridades.

2.3.4 PALAVRAS EMBUTIDAS EM VETORES

Para um computador processar alguma coisa, ela precisa estar adequada ao respectivo processamento. Um editor de texto comum lida com palavras e com formatação, mas não com ideias. Ele pode até conferir as palavras a partir de um dicionário ou verificar a consistência com as regras de escrita, mas não fará associações entre as palavras do texto. Partindo para a analogia com o ser humano, pode-se perguntar: como os seres humanos entendem uma frase ou palavra?

Ora, existem características comuns a todas as línguas escritas: são sequenciais; são entendidas a partir do significante (o texto escrito – elemento físico ou material) e do significado (conceitos inseridos de acordo com o contexto ou depreendido dele; elemento imaterial). Quanto ao significante, um dos problemas mais comuns é a questão dos homônimos, questão esta que não tem impacto direto na estrutura da rede neural artificial ou na eficiência da tradução. Porém, o significado de uma palavra pode ser afetado por elementos tais como a experiência pessoal, a razão, os conceitos e preconceitos. Elementos culturais podem influenciar também de forma indireta a partir da cultura, hábitos, comportamentos aceitos e ritos (VANIN, 2009). Se a língua tem influência na forma de pensar, seria viável a tradução? De acordo com Roman Jakobson (1959 *apud*. SCHERER; KADER, 2012, p.135):

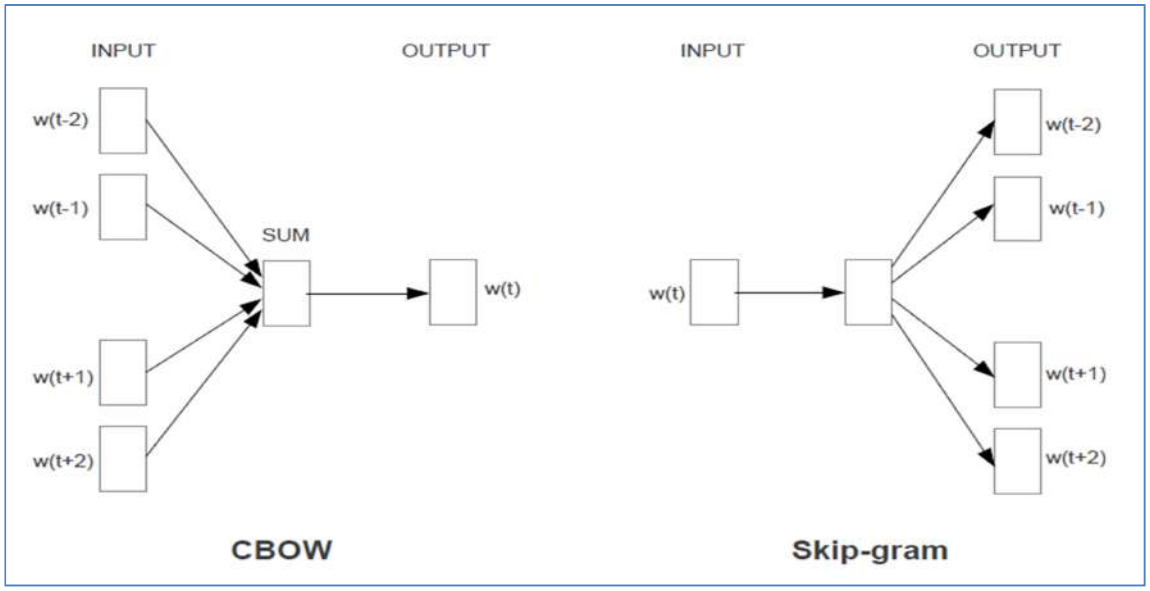
> Toda experiência cognitiva pode ser traduzida e classificada em qualquer língua existente. Onde houver uma deficiência, a terminologia poderá ser modificada por empréstimos, calcos, neologismos, transferências semânticas e, finalmente, por circunlóquios.

Essa afirmação aponta para a viabilidade da tradução. Conquanto o pensamento não dependa da linguagem para existir, no desenvolvimento humano seus caminhos se cruzam (VYGOTSKY, 2001). Mesmo pessoas que falam o mesmo idioma não trazem a mesma carga de significado para uma determinada palavra. Então, de certa forma, em toda a comunicação há tradução. A tradução em uma mesma língua é chamada de tradução intralingual, uma vez que o que recebe a mensagem interpreta o que a outra pessoa falou com o uso de palavras na mesma língua. A escolha das palavras levará em conta as relações mais fortes no contexto em que ela irá transmitir a mensagem. Da mesma forma, uma rede neural artificial deve ser capaz de apreender as relações entre as palavras usadas e saber encaixá-las na melhor ordem com o

objetivo de fazer a tradução.

Supondo que somente se quisesse transmitir uma lista contendo três palavras seguidas: gato, cachorro e bola. O vetor [1, 0, 0] pode representar o gato, o vetor [0, 1, 0] o cachorro e o vetor [0, 0, 1] a bola. Ora, cada nova palavra implicaria adicionar mais uma dimensão. Este modelo é usado para converter variáveis não numéricas em numéricas em aprendizagem de máquina e é chamado de *bag of words* (BOW), que significa literalmente uma bolsa de palavras. O resultado seria o acréscimo de mais uma dimensão para cada nova palavra, fazendo com que o número de dimensões seja igual ao número de palavras (MIKOLOV; YIH; ZWEIG, 2013).

Figura 4- CBOW e Skip-gram



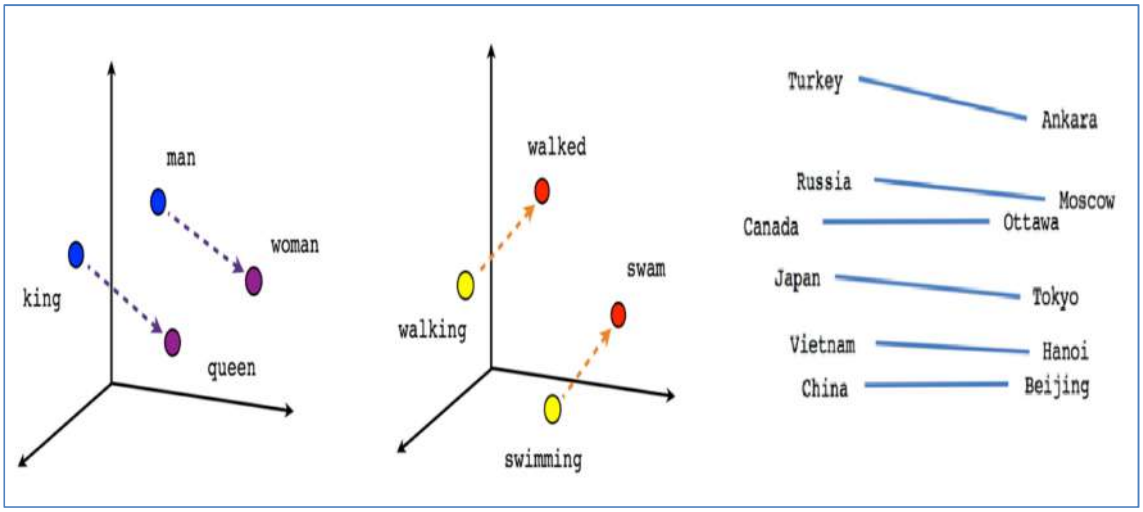
Fonte: MIKOLOV, 2013b.

O uso de técnicas de redução no número de dimensões é, portanto, um recurso extremamente útil para o processamento de linguagem natural. Porém, existem diferentes formas de se calcular os valores dos vetores. Duas delas são apresentadas na Figura 4. À esquerda está a CBOW (*Continuous Bag of Word*) que faz uso de uma janela limitando o número de palavras à frente e atrás em relação à palavra escolhida. Esse modelo deve predizer uma palavra a partir do seu contexto. Já no modelo *Skip-gram*, o objetivo é "maximizar a classificação de uma palavra com base em outra na mesma sentença" (MIKOLOV *et al*., 2013b).

A Figura 5, nos dois primeiros gráficos, mostra as relações lineares entre as palavras

e as distâncias correlatas na flexão de gênero e na flexão verbal. A projeção em duas dimensões, mais à direita, foi obtida a partir da transformação denominada de análise dos componentes principais (PCA) para poder apresentar vetores de 1000 dimensões em um gráfico plano (SMITH, 2002). Importa destacar que segundo Mikolov (2013a), os textos do treinamento não indicavam explicitamente a capital de cada país. As distâncias e os tipos de relações entre os nomes dos países e suas respectivas capitais é que propiciaram o resultado observado a partir das relações encontradas nos documentos utilizados. É possível, inclusive, fazer operações de soma e subtração a fim de obter algumas dessas relações.

Figura 5- Distância e direção vetorial das palavras



Fonte: MIKOLOV; YIH; ZWEIG, 2013.

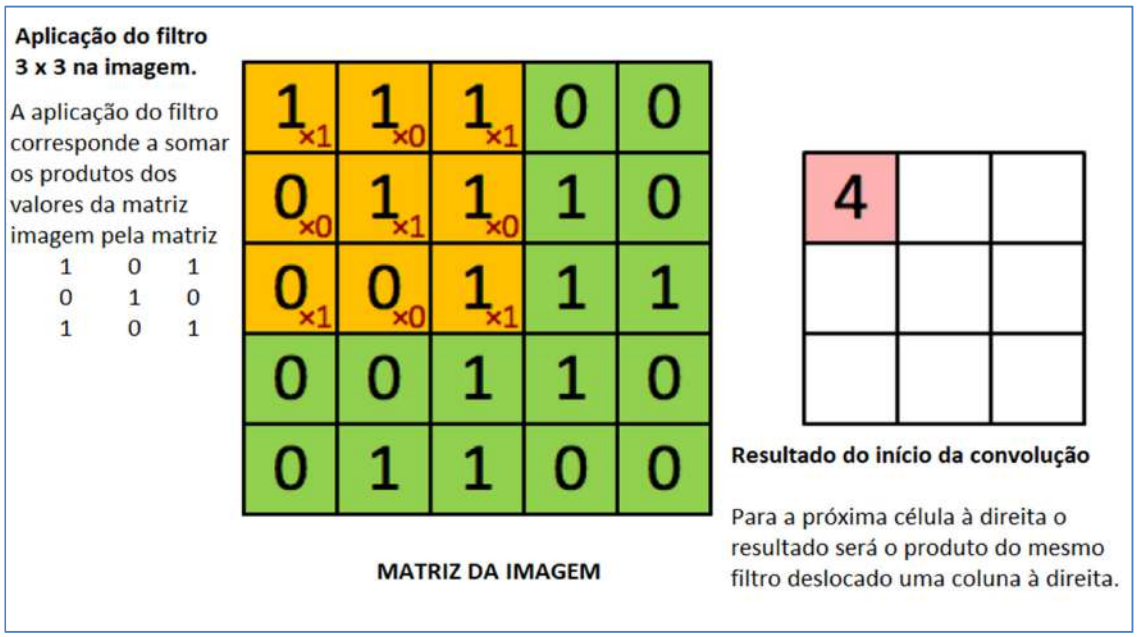
2.3.5 REDES NEURAIS CONVOLUCIONAIS

As redes neurais convolucionais, mais conhecidas como CNN (do inglês *Convolutional Neural Networks*) ou ConvNets, possuem origem no Neocognitron de Fukushima (1970), que se inspirou em estudos de células do córtex visual de gatos. O uso de redes neurais convolucionais é muito comum no reconhecimento de padrões e na computação visual. Porém, o uso de redes convolucionais no processamento de linguagem natural (NLP) vem ganhando espaço na tradução (KALCHBRENNER; GREFENSTETTE; BLUNSOM, 2014).

A Figura 6 mostra a operação de convolução aplicada em uma imagem. A região em amarelo corresponde à aplicação do filtro (também chamado de *kernel*) na matriz da imagem de dimensão 5x5. Como o filtro tem dimensão 3x3, ele poderá mover duas colunas para a

direita e duas para baixo, o que resultará em uma matriz 3x3. O valor da primeira linha é o resultado da soma dos produtos indicados nas células da matriz em laranja. Assim sendo, o ajuste no filtro proporciona resultados, transformações ou captura de propriedades distintas.

Figura 6- Parte do processamento de imagem de uma rede CNN



Fonte: STANFORD WIKI, 2018 (adaptado).

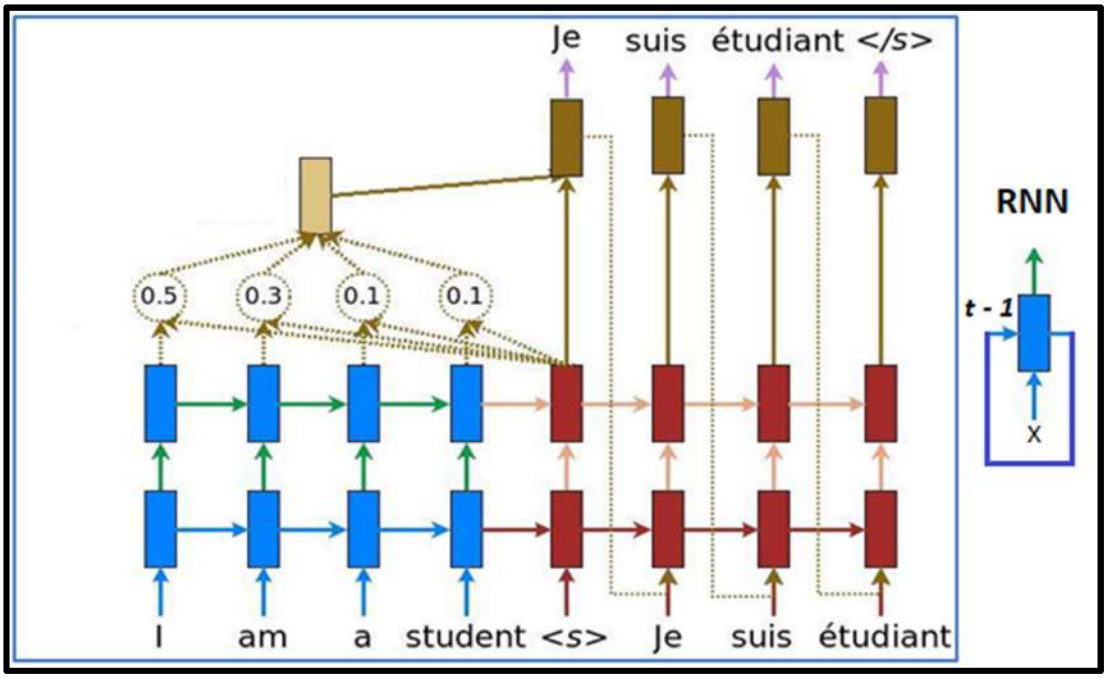
O projeto word2vec (WORD2VEC, 2013) é um exemplo do uso de redes CNN com o objetivo de representar palavras na forma de vetores. Nesse caso, cada palavra corresponde a uma linha da matriz. Assim, se cada palavra for transformada em um vetor de 200 dimensões, então, uma sentença de 15 palavras irá corresponder a uma matriz de 15 linhas por 200 colunas. O filtro, nesse caso, é utilizado para identificar propriedades da palavra ou suas relações com as demais palavras (KIM, 2018).

2.3.6 REDES NEURAIS RECORRENTES

As Redes Neurais Recorrentes, ou RNN (*Recurrent Neural Network*), levam em consideração o aspecto sequencial da linguagem. Por esse motivo, elas são muito utilizadas no processamento de linguagem natural, especialmente na tradução automática, uma vez que a linguagem também é sequencial. As variantes incluem chaveamento de palavras com base na

relevância entre elas no contexto em que estão inseridas (CEPERO, 2018).

Figura 7- Rede Neural Recorrente



Fonte: CEPERO, 2018 (adaptado pelo autor).

Considerando as entradas de um neurônio artificial, a recorrência consiste em adicionar como entrada no processamento da célula RNN a saída do intervalo de tempo anterior, como indicado à direita na Figura 7. Isso significa que a saída em (t – 1) é mais uma entrada da rede em t. Porém, a forma mais fácil de visualizar uma rede neural recorrente é dela aberta no tempo, como à esquerda na Figura 7, desenrolando a rede repetindo a célula para cada momento no tempo para cada termo. Assim é possível mostrar cada palavra em sequência como entrada ou saída da rede.

Os retângulos em azul na Figura 7 correspondem às células RNN de codificação, responsáveis por extrair uma representação vetorial da sentença na língua de entrada. Esta representação é chamada de vetor de contexto. Por outro lado, os retângulos em vermelho correspondem às células RNN de decodificação, e são responsáveis por converter a representação da sentença em uma sentença na língua de saída do tradutor. Porém a camada de codificação também pode receber informações do decodificador com a finalidade de acelerar a aprendizagem da rede e melhor avaliar a escolha do resultado da tradução. Em redes RNN básicas, o vetor de contexto possui tamanho fixo, ainda que a sentença possua

tamanho variável (CHO, 2014). Contudo, uma camada adicional permite selecionar as partes mais relevantes da sentença e promover o alinhamento automático para a tradução na língua destino a partir do vetor de contexto, que, em redes RNN não precisa ser fixo (BAHDANAU, 2014). Os retângulos em bege e marrom correspondem a essa camada. O vetor que segue do retângulo bege para o marrom é chamado de vetor de atenção (*attention vector*), cujo objetivo é aprimorar a escolha final das palavras levando em conta o contexto.

Na parte inferior da Figura 7, as células em azul representam o codificador e as em vermelho o decodificador. Essa estrutura também é chamada de codificador-decodificador. No caso da Figura 7, o codificador recebe e aprende a processar sentenças em inglês, enquanto que o decodificador aprende a traduzi-las para o francês. As marcações <s> e </s> representam início e fim de cada sentença. Em alguns casos a marcação do início e fim de sentença é representada por <bos> e <eos>, respectivamente. O mais importante nesse momento é perceber que após receber a palavra "*I*", em inglês, a célula produz uma saída para a célula da próxima camada de neurônios acima dela e também para a célula seguinte da mesma camada que ela. As saídas podem ser iguais ou não, dependendo do tipo de rede RNN. O processamento da palavra seguinte, que neste caso seria a palavra "*am*", em inglês, será afetado pela saída da célula anterior. Fica clara a dificuldade de se aplicar o processamento em paralelo em redes recorrentes, uma vez que há dependência dentro de uma mesma camada e a primeira palavra acaba por ter maior influência no processamento da sentença das demais (GEHRING *et al*., 2016). Contudo, o problema mais grave é a atenuação acentuada do gradiente em sentenças muito longas. Dentre as soluções estão:

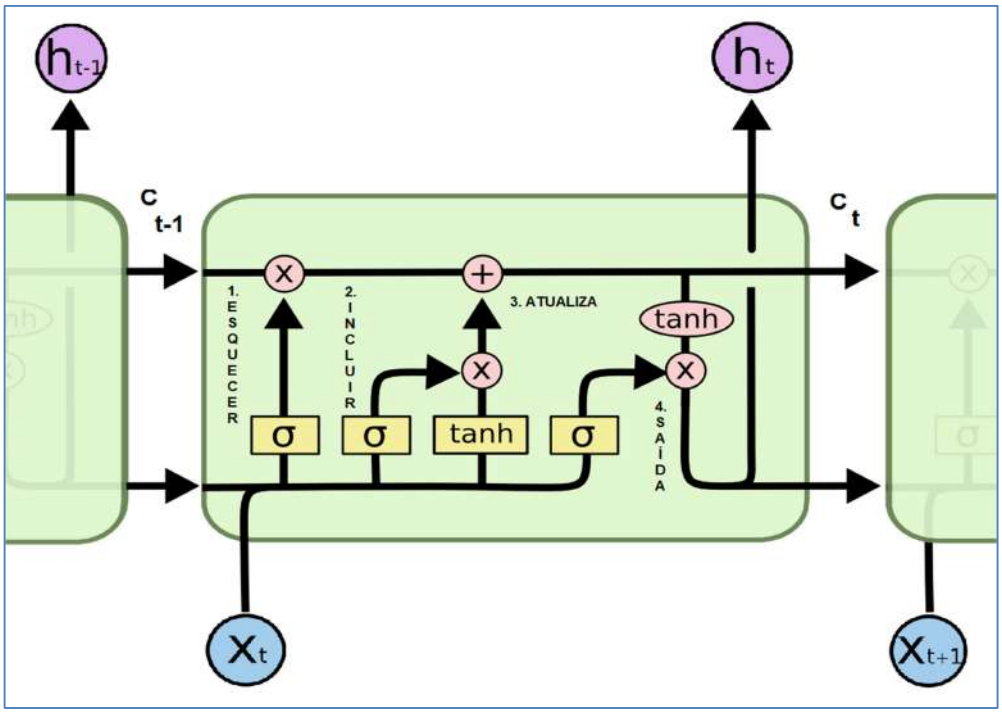
- Uso da ReLU como função de ativação (uma vez que ela não achata valores positivos de entrada).
- Aumento do processamento em lote (*batch*) para o cálculo do gradiente, o que demanda mais memória.
- Uso de conexões adicionais entre as camadas fazer uso de uma das variantes da rede neural recorrente, tais como a LSTM.

2.3.7 REDES NEURAIS RECORRENTES LSTM

A rede neural recorrente LSTM (*Long Short Term Memory*) é uma variante da rede RNN. A Figura 8 mostra a estrutura interna de uma célula recorrente LSTM, em que o

símbolo **σ** (sigma grego) que representa a função de ativação logística, também chamada de sigmoide, conquanto o gráfico da função tangente hiperbólica também seja semelhante a uma sigmoide. Os símbolos nos círculos representam as chaves usadas na LSTM. Por meio dessas chaves, a rede LSTM é capaz de controlar até que ponto algumas palavras devem ser "esquecidas", "lembradas" ou "ignoradas" (HOCHREITER; SCHMIDHUBER, 1997).

Figura 8- Estrutura LSTM



Fonte: OLAH, 2015 (adaptado pelo autor).

Termos identificados como menos relevantes serão "esquecidos", o que é feito pela chave junto à primeira função logística na Figura 8, e substituídos por um mais relevante (chaves de inclusão seguida da atualização). A manutenção em memória não depende de uma janela fixa, mas da relevância das relações dos termos na sentença. Quando o termo for relevante, a chave tenderá a 1, indicando que deve ser mantido (não esquecido), ou incluído e atualizado em "memória" (MARUF; HAFFARI, 2017). O mesmo ocorre na seleção da saída para a próxima camada, em que 1 significa passar para a próxima camada e 0 significa não passar (BAHDANAU; CHO; BENGIO, 2014).

A via superior da Figura 8 é por onde passa a memória atualizada da célula LSTM ($C_{t-1} \rightarrow C_t \rightarrow C_{t+1}$), ou estado da célula LSTM, que não é passado diretamente para a próxima camada. A saída para a próxima camada é controlada pela última função logística, que

funciona como uma chave por multiplicar os valores por zero ou 1. As letras h_{t-1} e h_t para essas saídas fazem referência às camadas escondidas (*hidden layers*) nos tempos *t*-1 e *t*. Existem variantes de células recursivas com chaves com poucas diferenças do que foi apresentado, como é o caso da GRU (*Gated Recurrent Unit*) e da SRU (*Simple Recurrent Unit*). Ambas são mais rápidas que a LSTM, contudo as redes com células LSTM respondem melhor a um grande volume de dados e a sequências maiores, caso em que as camadas possuem um grande número de células recorrentes.

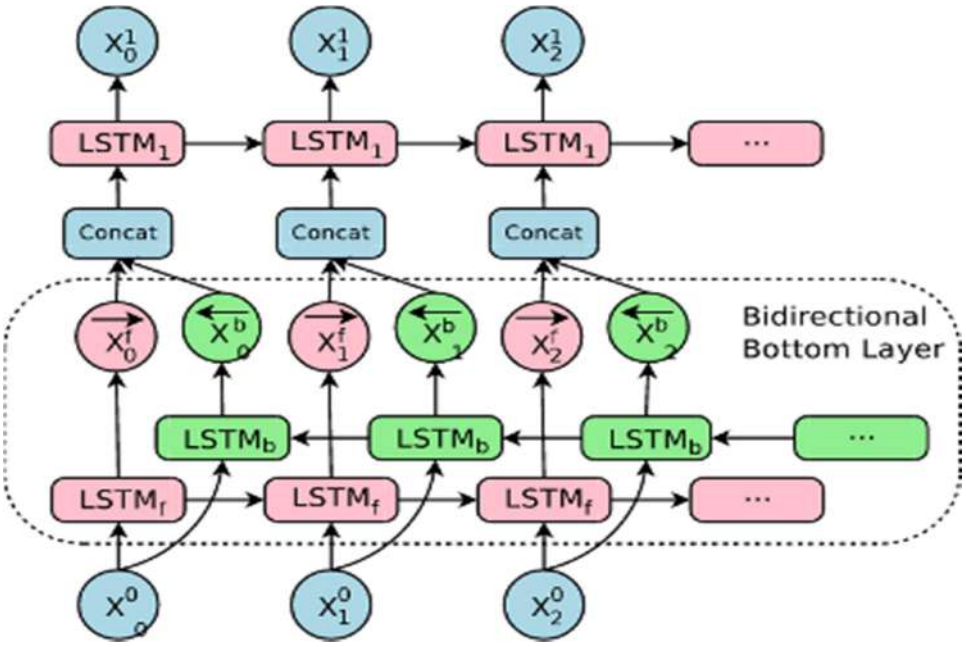
2.3.8 REDES RECORRENTES BIDIRECIONAIS

Um dos motivos para a construção de redes recursivas bidirecionais é atender à questão levantada sobre o desequilíbrio na codificação dos termos da sentença. Especialmente na codificação, em uma camada unidirecional, o último termo tem pouca influência na captura da informação. Por outro lado, o primeiro termo irá influenciar o processamento de todos os demais termos da sentença.

A solução proposta é a de dividir a primeira camada de codificação em duas partes, cada uma em um sentido. A primeira metade segue a partir do início da sentença, enquanto que a segunda metade segue de trás para frente, a partir do último termo da sentença. Os resultados são então concatenados e passados para a próxima camada. A camada bidirecional permite, portanto, que sejam considerados os dois sentidos da sentença. O modelo apresentado na Figura 9 é o de uma rede LSTM semelhante ao que fora adotado pela equipe da Google (WU *et al*., 2016). Para se entender a figura, é importante observar que a entrada segue de baixo para cima, de forma que a camada inferior é bidirecional (*Bidirectional Bottom Layer*).

As entradas da primeira camada da Figura 9 estão assinaladas com o zero sobrescrito, isto é, x_0^0, x_1^0 e x_2^0. Por outro lado, x_0^1, x_1^1 e x_2^1 correspondem à saída da segunda camada LSTM, que não é bidirecional. A função Concat concatena os vetores resultantes de ambos os sentidos da rede bidirecional e fornece a entrada da segunda camada escondida. Ao invés da concatenação dos valores, os mesmos podem ser somados (BRITZ *et al*., 2017).

Figura 9- Codificador LSTM bidirecional



Fonte: OPENNMT.NET (2018)

Ao se usar o modo bidirecional com concatenação, se o codificador da rede for definido com 800 entradas, por exemplo, a primeira camada será dividida em duas partes de 400 entradas, cada uma lerá a sentença em uma direção. As demais camadas não são bidirecionais e recebem o resultado da operação, que no caso seria a concatenação.

No uso padrão da LSTM não bidirecional, o número de entradas do idioma fonte seria, no exemplo dado, de 800 em um único sentido. Um aspecto considerado negativo de utilizar o modelo bidirecional é a maior dificuldade de paralelizar o processamento de dados (WU *et al*., 2017).

2.3.9 PONTES E OUTRAS CONEXÕES

Conexões entre diferentes pontos de uma rede neural artificial podem ser criadas com o objetivo de acelerar a inicialização dos pesos das entradas das células artificiais, evitar a degradação da aprendizagem, normalizar valores ou evitar a perda de informação de controle geral da rede. A *bridge* (ponte) é usada para compartilhar o estado final do

codificador e o início do decodificador, mas pode ter poucos resultados se depender apenas de sua implementação. A concatenação do vetor de contexto como uma entrada adicional para o decodificador pode ter efeitos similares ao uso da ponte em redes recorrentes na aceleração do aprendizado (AGRAWAL, 2017).

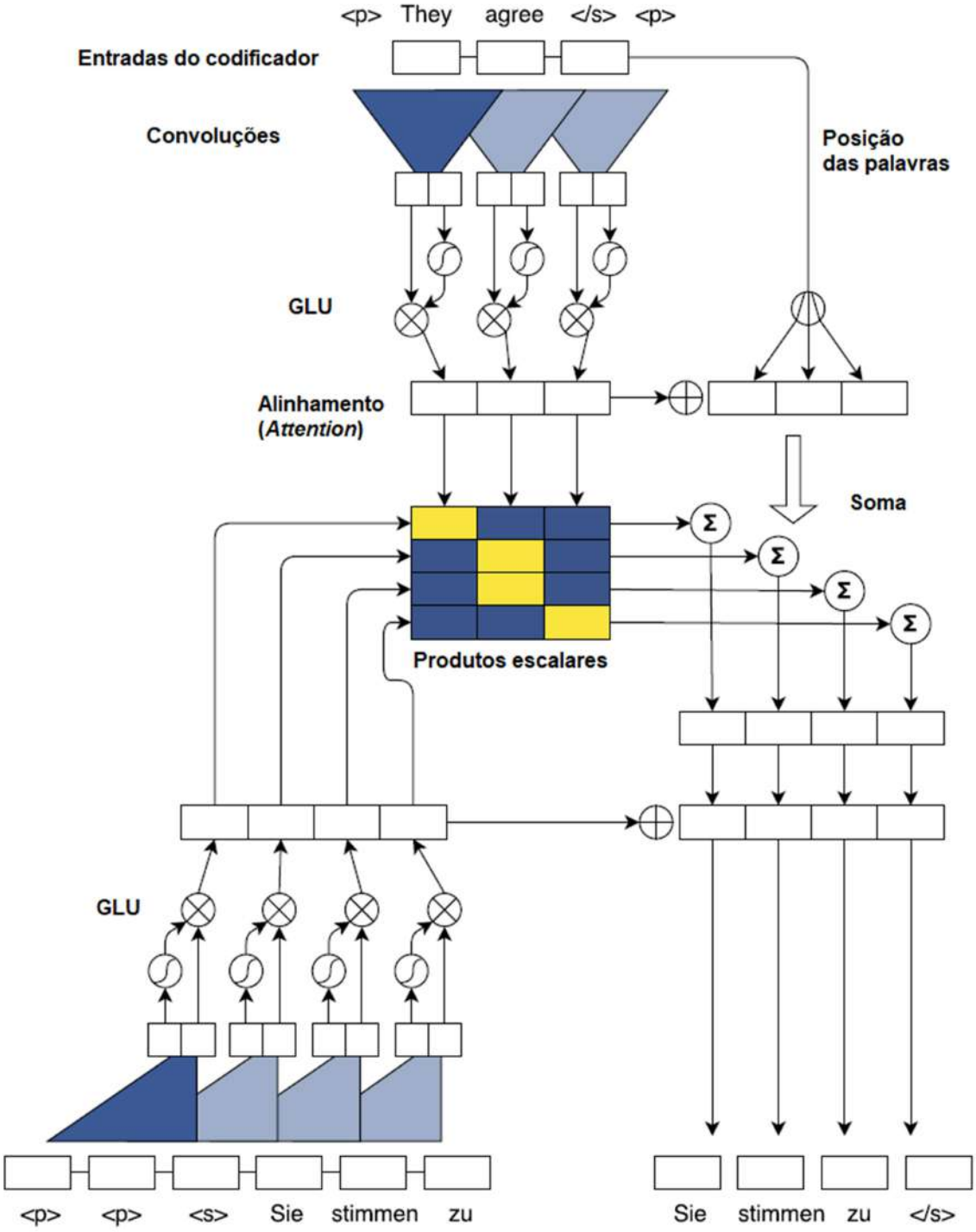
O uso de conexões residuais, que reforçam ou inibem os efeitos da uma entrada sobre a saída correspondente com o uso de outro tipo de função de ativação, pode evitar a degradação do gradiente na atualização dos pesos em redes recorrentes com muitas camadas (geralmente mais de quatro). Já em redes CNN, por seu perfil não sequencial e mais linear, quanto maior o tamanho da sentença, mais difícil será controlar o contexto e o alinhamento da tradução. As conexões residuais servem para levar informações das entradas, antes de cada convolução, para a saída. A aplicação de uma função não linear de ativação permite um melhor controle do tamanho da sentença e do correspondente vetor de contexto (GEHRING *et al*., 2016). A adaptação das conexões pode exigir a criação de uma camada específica de normalização com o objetivo de estabilizar e acelerar o treinamento da rede (ZAREMBA; SUTSKEVER; VINYALS, 2014). Propostas de modelos híbridos procuram reunir os pontos fortes de cada tipo de rede (BA; KIROS; HINTON, 2016; CHEN; FIRAT; BAPNA, 2018).

2.3.10 REDES CONVOLUCIONAIS CONVS2S

Como já explanado, redes RNN levam desvantagem em relação a redes CNN quando se trata de processamento em paralelo. Por outro lado, redes CNN levam desvantagem em relação a redes RNN quando se trata de sentenças longas. A preocupação em se adequar ao processamento em paralelo tem por justificativa o desenvolvimento do hardware. Como em redes RNN, o próximo estado depende do estado anterior da camada escondida, não apenas da entrada, há uma cadeia de dependência que prejudica o processamento em paralelo. A principal solução encontrada para redes RNN em sequências mais longas foi com o uso de chaves, como ocorre nas redes LSTM. A Figura 10 apresenta o modelo ConvS2S, nome atribuído ao modelo proposto por Yann Dauphin, Angela Fan, Michael Auli e David Grangier (2017) que faz o uso de chaves não recursivas após as camadas de convolução, com a obrigação de que nenhuma convolução contenha valores com dependência temporal entre si. Essas chaves foram por eles denominadas GLU (*Gated Linear Units*), uma vez que tais chaves não eram mais recorrentes e controlavam a passagem de informações que resultavam

de operações lineares (DAUPHIN *et al.*, 2017).

Figura 10- Treinamento de uma rede ConvS2S



Fonte: GEHRING *et al.*, 2017 (adaptado pelo autor).

Para garantir a referência às posições dos termos, foi incluída uma coluna com a posição absoluta de cada termo da sentença nos vetores que representam as palavras na sentença, ao que se denomina embutir o posicionamento no vetor (*positioning embeddings*). A saída que se segue à convolução é concatenada com conexões residuais (região que antecede as GLU na Figura 10) e as funções de ativação usadas nas chaves GLU são funções logísticas. A finalidade é desconsiderar os elementos desnecessários à tradução. Após a camada de codificação da entrada é considerado o cuidado com o alinhamento e contexto, o que inclui a correção no posicionamento das palavras no texto (LUONG; PHAM; MANNING, 2015). Portanto, a soma dos dados de posicionamento dos termos na sentença reforça a escolha dos mesmos. Ao final, os cálculos levam em conta, simultaneamente, o posicionamento, as saídas do codificador e do decodificador (GEHRING *et al*., 2017).

A preocupação com a normalização se reflete tanto nas conexões residuais, como na inicialização, e visam tornar a aprendizagem mais estável. O desvio padrão é ajustado de acordo com os módulos da rede geral. Nas entradas, por exemplo, o desvio padrão recomendado é de 0,1, enquanto que para as camadas que não repassam sua saída diretamente para as GLU, o valor corresponde a $\sqrt{1/n}$, em que *n* corresponde ao número de conexões de entrada do neurônio (GEHRING *et al*., 2017).

Os resultados obtidos com as redes ConvS2S garantiram sucesso às redes convolucionais com sentenças mais longas, ainda com certa limitação. No processamento com o uso de CPU, as redes ConvS2S chegaram a superar os resultados obtidos com redes LSTM em eventos. Uma possível justificativa é a de que empilhamento de camadas convolucionais possa refletir melhor a estrutura hierárquica da linguagem.

2.3.11 MODELO TRANSFORMER

No modelo de rede neural denominado Transformer, a camada de atenção torna-se protagonista. Dependendo da configuração ela pode até ser uma simples função *Softmax*, como inicialmente foi entendida em redes recorrentes. Afinal, a camada de atenção deve ser capaz de colocar o foco naquilo que é relevante a partir de informações dadas e, se existem informações suficientes sobre posicionamento e contexto, não há muito que fazer. Por outro lado, as redes concorrentes sequenciais obtiveram sucesso com o posicionamento embutido e

o uso das conexões residuais para fins de contexto, pelo que maior atenção passou a ser dada para as conexões e para a camada de atenção. Seguindo esse caminho, verificou-se que identificar a melhor sequência com os termos mais relevantes de acordo com o contexto é o que basta para a tradução. Assim sendo, o modelo Transformer não é nem recorrente, nem convolucional, e conta com camadas de atenção mais elaboradas que assumem o papel na codificação e decodificação.

As camadas de codificação e decodificação possuem subcamadas, em que uma delas funciona como camada interna de atenção, também chamadas de *intra-attention* ou *self-attention*. Codificador e decodificador possuem o mesmo número de camadas e em ambos o posicionamento das palavras na sequência é embutido. Ambos iniciam com uma camada de atenção e com conexões residuais, as quais repassam informações normalizadas da entrada na saída. Contudo o decodificador possui uma subcamada a mais intermediária, que agrega a saída do codificador, totalizando três subcamadas. Os dados que chegam para a camada intermediária do próprio decodificador são chamados de *queries*, enquanto que os dados provenientes da saída da camada de codificação são chamados de chaves e valores. A menos dos nomes dados, esta configuração também pode ser encontrada na camada de atenção das redes ConvS2S (VASWANI *et al*., 2017).

Uma das principais mudanças em relação aos demais modelos foi o uso do mecanismo chamado de *multi-head attention*. Cada "cabeça" (*head*, em inglês) pode cuidar de subespaços distintos, os quais correspondem a contextos distintos em diferentes posições. O processamento por camada é feito em paralelo com as matrizes correspondentes às queries, chaves e valores e considera a relevância de acordo com os subespaços dos vetores, o que reduz o custo de processamento (VASWANI *et al*., 2017).

2.4 ASPECTOS LINGUÍSTICOS NA TRADUÇÃO AUTOMÁTICA

O desenvolvimento da tradução automática reúne saberes de diferentes áreas. As seções deste tópico abordam os aspectos linguísticos associados a elementos cognitivos da linguagem, formação das palavras e significado das mesmas.

2.4.1 DICIONÁRIO E VOCABULÁRIO

O primeiro uso efetivo de um mecanismo de tradução (ou cérebro eletrônico) foi registrado em 1954 com o objetivo de fazer uma tradução do russo para o inglês. O mecanismo contava com baixa capacidade de armazenamento de palavras, pelo que foi utilizado um micro glossário (BOOTH, 1954). Segundo Hutchins, apesar do pequeno número de palavras utilizado na tradução, o evento, um resultado da cooperação entre a Universidade de Georgetown e a IBM, foi suficiente para impressionar o público e motivar a pesquisa em busca de ferramentas de tradução (HUTCHINS, 2017).

Dicionários, léxicos ou glossários, geralmente não contém todo o vocabulário, motivo pelo qual algumas palavras foram escolhidas para representar outras que possuem o mesmo radical e significado muito próximo. Tais palavras são chamadas de lemas. A ideia é reduzir a quantidade de palavras e ainda poder recuperar seu significado. Na língua portuguesa, o infinitivo impessoal de um verbo é usado como o lema de todas as suas flexões. O particípio de um verbo também pode ser encontrado quando usado como adjetivo. No caso de substantivo e adjetivos, geralmente a preferência recai sobre a forma singular e masculina (caso flexione em gênero). Então, “estudar” é lema para “estudaram” e “estudei”, enquanto que “menino” é lema para “meninos”, “menina”, “meninice” e “meninada”.

Antes do efetivo treinamento da rede neural artificial é feita uma lista com todas as palavras, números e sinais presentes nos arquivos de texto e a frequência com que aparecem. Essas informações irão compor o vocabulário. Contudo, quanto maior o vocabulário, maior o consumo de memória. Por isso, é comum impor limites ao vocabulário, seja pelo número de elementos, seja pela frequência mínima em que aparecem os termos, a fim de guardar apenas o que é mais relevante na tradução (BAHDANAU; CHO; BENGIO, 2014).

Algumas formas de se reduzir o vocabulário incluem:

- Usar as palavras em minúsculo sem perder informação. Como o uso de uma palavra em maiúsculo ou minúsculo pode mudar seu sentido. O uso de códigos para representar as propriedades da palavra original é uma forma de manter a informação.
- Limitar a inclusão de palavras no vocabulário a uma frequência mínima de incidência nos textos. Pode-se forçar a inclusão de palavras no vocabulário com a repetição dos textos em que elas aparecem. No caso de se aumentar o número de

passos de um treinamento, caso haja interesse na utilização de novas palavras, recomenda-se manter as anteriores.

- Limitar a inclusão de palavras no vocabulário a uma determinada quantidade de termos. Nesse caso, os termos serão contados a partir da ordem decrescente segundo a frequência em que eles ocorrem.
- Restringir apenas a palavras, ou a palavras e sinais, dentro de uma faixa de caracteres aceitos. Assim, termos que não forem formados apenas por letras e sinais, serão desconsiderados.
- Converter termos codificados, tais como endereços de internet, números, valores em moeda etc. Endereços de internet podem ser convertidos em _url_ e podem ser seguidos de números. Contudo, fica claro que tais valores precisam ser guardados para substituição na decodificação a fim de serem substituídos pelos termos originais.
- Corte ou fracionamento de palavras. Técnica conhecida como *steeming*. As palavras são recortadas com o objetivo de remover os afixos, sobrando geralmente o radical. É de pouca utilização para a tradução.
- Segmentação de palavras. Uma palavra pode ser vista como uma sequência de pedaços de palavras. Existem diferentes técnicas para o fracionamento de palavras e, no geral, a principal vantagem é a resposta a palavras desconhecidas, uma vez que o fracionamento aumenta o número de combinações de pedaços de palavras.

2.4.2 ESTRUTURA E FUNÇÃO DAS PALAVRAS

Outro aspecto linguístico importante é o da formação e das funções das palavras. Na língua portuguesa, por exemplo, dentre os elementos que conectam orações ou outras palavras, escritos destacados destas, e que trazem informações sobre as relações entre elas, encontram-se preposições, conjunções e pronomes. Pronomes e artigos, indefinidos ou não, podem trazer informações de identificação, gênero e número. Outros elementos destacados, não unidos a uma palavra, atribuem intensidade, tempo, lugar, em suma, complementam as demais informações.

Porém, relações implícitas na formação das palavras podem parecer simples para um

ser humano, mas não para um computador (MARKIEWICZ; ZHENG, 2017). Na língua portuguesa, algumas preposições podem se ligar a um artigo ou pronome, como é o caso de “em” + “os”, que se torna “nos”, e também de “a” + “aquela”, que se torna “àquela”. Situação similar ocorre com morfemas (unidades linguísticas menores que mantém algum significado) que se ligam, com ou sem hífen, na formação de palavras, como é o caso de prefixos e sufixos. As desinências, como sufixos que indicam flexões verbais (modo, tempo, pessoa e número) ou nominais (gênero e número), acabam integrando a palavra. Em tais casos, a identificação automática da estrutura gramatical está implícita ou sintetizada em uma única palavra, o que aumenta a complexidade da solução (CHO; MERRIËNBOER; BAHDANAU, 2014).

2.4.3 FRACIONAMENTO

A primeira máquina de tradução fez uso de um processo denominado stemming, processo em que são usados apenas os pedaços invariantes da palavra a fim de demandar menos processamento (BOOTH, 1954). Descobrir em que pedaços separar é algo que se pode encontrar em um corpus, ou conjunto de textos em formato eletrônico com informações relativas a uma língua específica. Outra forma é fazer uso de modelos estatísticos. Estes pedaços podem ser iguais aos radicais da palavra. No caso das palavras “estudar”, “estudante” e “estudioso”, o *stem* de todas elas é “estud”. Trata-se de um processo pouco recomendável na tradução porque muita informação relevante é perdida.

2.4.4 COMPACTAÇÃO

Um vocabulário muito grande no treinamento vai impactar em maior consumo de memória e menor desempenho computacional. A ideia da compactação é reduzir o tamanho do significante, que no caso de um documento na forma de texto é a palavra escrita. O significado é mantido, e revelado na descompactação. De certa forma, a substituição de termos que não variam de acordo com o idioma, tal como um endereço de e-mail, por códigos numerados pode ser vista como uma forma de compactação. Porém, a forma de compactação mais utilizada é a substituição de caracteres que se repetem por um caractere que não seja

encontrado nos termos do vocabulário, ou ainda um símbolo.

A técnica de compactação mais usada atualmente é chamada de BPE (*Byte Pair Encoding*), que consiste em substituir um par de caracteres que se repete por um caractere não utilizado. O projeto *subword-nmt* implementou a compactação BPE (SENNRICH; HADDOW; BIRCH, 2015). Porém, algumas bibliotecas de software, cujo objetivo é fazer a tokenização, também incluíram BPE como uma de suas funcionalidades. As bibliotecas "*sentencepiece*" e "*pyonmttok*", ambas usadas no projeto OpenNMT-py do *Github*, incluem BPE.

A fim de poder reverter o processo de compactação, deve haver pelo menos um marcador do fim do termo (geralmente </w>) e devem ser guardados os códigos usados na substituição. Se a segmentação for por caractere (LING *et al*., 2015), então, o vocabulário irá conter apenas símbolos, letras, números e sinais, tornando-se bastante pequeno. Se por um lado a segmentação pode diminuir o número de termos no vocabulário, por outro lado, o tamanho da sequência terá de ser aumentado, o que representa um maior número de entradas (MIKOLOV; YIH; ZWEIG, 2013), o que ainda pode ser compensador, especialmente quando as línguas de origem e destino fizerem uso do mesmo alfabeto.

2.4.5 SEGMENTAÇÃO N-GRAM

Semelhante à compactação, o objetivo é consumir menos memória a fim de viabilizar um melhor desempenho computacional. Contudo, na segmentação, não há substituição de caracteres. Cada palavra é segmentada em certo número de caracteres. O termo n-gram se refere a uma sequência de um número *n* de itens de uma amostra de um texto. No caso da segmentação de palavras, o termo se refere ao número de caracteres. A segmentação de palavras caractere a caractere corresponde à segmentação n-gram = 1. Quanto maior a janela de caracteres que irá percorrer a palavra, menor o número de segmentos. Por outro lado, quanto menor a janela de caracteres, menor o vocabulário. A segmentação de palavras por caractere é recomendada quando as línguas de entrada e de saída possuem origem e caracteres comuns (TIEDEMANN, 2012). Se as proximidades entre as línguas for grande, recomenda-se o compartilhamento de um único vocabulário.

2.4.6 ASPECTOS COGNITIVOS E ZERO-SHOT

Muitas ideias surgiram para suprir a falta de *corpus* com textos em paralelo. Dentre elas se destaca a linguagem intermediária. Uma terceira língua que possua fontes de dados alinhados com a língua de entrada e com a língua de saída. A equipe liderada por Nikolaj Andreev da Universidade do Estado de Leningrado, Rússia, considerou a ideia de construir uma linguagem artificial intermediária concreta e completa. Essa ideia está próxima do chamado Sonho de Leibniz: uma linguagem universal capaz de expressar os pensamentos de forma clara, sem equívocos ou ambiguidades. Uma linguagem capaz de "estreitar a distância que separa um Deus que criou o Universo pela palavra e o homem que constrói um universo de palavras" (POMBO, 1997, p.261).

A técnica denominada *Zero-shot* faz uso de línguas intermediárias a fim de construir uma ponte entre línguas que possuem pouco ou nenhum material para treinar um tradutor automático (JOHNSON *et al*., 2016). Contudo, o uso de uma língua intermediária que não seja rica em elementos que facilitem a identificação dos tempos verbais, número, gênero, sujeito da ação, complementos e outros elementos textuais, acaba por empobrecer a tradução.

Na língua portuguesa, por exemplo, informações são anexadas às palavras indicando: número e gênero de artigos, pronomes, substantivos e adjetivos; tempo, modo e pessoa na flexão verbal; graus de advérbios; e função sintática em pronomes pessoais. Línguas como o Latim, por exemplo, além do gênero neutro e das flexões verbais, possuem flexões de casos para substantivos e adjetivos, as quais identificam a função sintática da palavra na frase.

Isso não significa que as demais línguas não tenham outras formas de se identificar os casos elencados. No caso da língua portuguesa, preposições, pronomes, posicionamento das palavras e sinais de separação (tal como a vírgula) cumprem esse papel. Acontece que o papel de uma preposição depende dos outros termos associados a ela. Em algumas línguas, um artigo pode ser modificado devido ao efeito fonético que resulta de sua associação com a palavra seguinte, o que dificulta o trabalho do tradutor. Esses são alguns dos grandes desafios enfrentados pelo uso de uma linguagem intermediária.

2.4.7 GLOSSÁRIO OU BASE DE TERMOS ESPECIALIZADOS

A tradução dentro de uma sublinguagem específica reduz o tamanho do vocabulário utilizado, contudo em se tratando de documentos legislativos de diferentes países, existe a necessidade de passar ao tradutor a origem do documento que será traduzido. É recomendável que os termos de uso específico presentes em tais documentos sejam identificados, ou protegidos, para fins de tradução direta. Em visitar à Assembleia da República de Portugal, foi possível verificar grandes diferenças no uso de termos legislativos.

Quadro 1- Base de Termos da Assembleia da República de Portugal

Português	Inglês	Francês
proposta de lei	Government bill	projet de loi
projeto de lei	members' bill	proposition de loi
projeto de resolução	draft resolution	proposition de résolution
projeto de revisão constitucional	draft amendments to the Constitution	proposition de révision constitutionnelle

Fonte: PORTUGAL (2019).

O Quadro 1 apresenta parte da base de termos da Assembleia da República de Portugal. Nele está destacado que "proposta de lei" e "projeto de lei" recebem a qualificação da origem do documento na tradução em Inglês. Contudo a tradução em Francês parece invertida, visto que "projeto" não é traduzido como "*projet*", enquanto que "proposta" e "proposição" são traduzidas por "*proposition*". Por outro lado, o termo "*draft*" é usado como tradução de "projeto", em Português, quando não associado ao termo "lei". Tais correspondências não são necessariamente válidas para países fora da União Europeia. Este é o caso do Brasil. Isso mostra a importância do uso de glossários ou bases de termos legislativos para a pesquisa interparlamentar.

Considerando o que representa tal constatação, foi solicitado ao Grupo de Trabalho pela Integração entre as casas legislativas do Congresso Nacional, Câmara dos Deputados e Senado Federal, a tradução do Glossário de Termos legislativos do Congresso Nacional nos

seguintes idiomas: Inglês, Francês e Espanhol. A equipe de tradução do Senado Federal tomou a iniciativa de traduzir também para o Italiano.

O Quadro 2 trata-se de uma amostra da tradução do glossário em curso que foi elaborado pelo Serviço de Tradução do Senado Federal. A partir da segunda coluna, tem-se a tradução dos termos para o Inglês, Francês e Espanhol. A tradução do glossário está prevista para agosto de 2019.

Quadro 2- Tradução do Glossário do Congresso Nacional

Lista de conceitos	List of concepts	Liste des concepts	Relación de conceptos
Abertura de Reunião	**Opening of Meeting**	**Ouverture de réunion**	**Apertura de la reunión**
Ato do presidente de comissão que dá início a uma reunião da comissão.	Act of the Commission Chairman who initiates a commission meeting.	Acte du président de commission ouvrant une réunion de celle-ci.	Acto del presidente de la comisión por el cual se abre una reunión de comisión
Ver também:	*See also:*	*Voir également*	*Ver también*
Quórum de Abertura de Reunião *e* Reunião.	Meeting Opening Quorum *and* Meeting.	Quorum d'ouverture de réunion et Réunion.	Quórum de aperture de la reunión y la reunión.
Abertura de Sessão	**Opening of Session**	**Ouverture de séance**	**Apertura de la sesión**
Ato do presidente que declara abertos os trabalhos da sessão plenária.	Act of the President declaring open the proceedings of the plenary session.	Acte du président déclarant ouverte la séance plénière.	Acto del presidente por el que se declaran abiertos los trabajos de la session plenaria.

Fonte: BRASIL (2019).

No caso da tradução do idioma de um país para outro idioma sem fazer uso da identificação da origem, pode-se incluir o glossário ou a base de termos repetidas vezes de modo a garantir a maior frequência dos termos corretos. Outra opção é pesquisar os termos, protegê-los e fazer a substituição direta na saída do tradutor. A solução irá depender do processo utilizado antes do tradutor.

No caso de se fazer a identificação da origem, cada início de linha deve incluir uma marca que identifica o país de origem do texto. Tal marca deve chegar ao tradutor e deverá influenciar a tradução de toda a linha.

2.5 PONTO DE PARTIDA PARA A CODIFICAÇÃO

O desafio de desenvolver uma solução de tradução automática para uso da Câmara dos Deputados em menos de um semestre a partir do zero não se mostrou viável, ainda mais que nos primeiros meses sequer havia equipamento adequado. Em razão disso, foram

comparadas as características gerais de projetos abertos que já haviam mostrado algum resultado. Projetos compatíveis com outros projetos já em curso eram preferíveis. Na época estava sendo desenhado um projeto de reconhecimento de orador por meio de redes neurais que seria construído em Python e que faria uso de algumas bibliotecas abertas já selecionadas.

O projeto de rede neural OpenNMT foi escolhido pelas seguintes razões:

- ter o código aberto com três grandes linhas de desenvolvimento divididas por linguagem de programação e bibliotecas de software (Lua e Torch; Python e PyTorch; Python e TensorFlow);
- ser um projeto ativo com atualizações frequentes de correção e de inclusão de novas técnicas de tradução utilizadas em eventos internacionais de tradução automática;
- ser modular e adaptável;
- não se limitar a entradas textuais, mas incluir entradas em áudio e imagem;
- ter mais de um tipo de implementação, com Torch ou TensorFlow, fazer uso de linguagens conhecidas ou simples, tais como Python e Lua.

O projeto OpenNMT partiu de uma iniciativa de desenvolvimento em código aberto, padrão MIT, modelos de redes neurais visando a tradução (KLEIN *et al*., 2017). O padrão de código aberto do MIT é permissivo, mas exige que seja mantida uma cópia da licença junto ao código aberto. O projeto OpenNMT recebe colaboração tanto acadêmica, contando com voluntários de todo o mundo, como da parte de grandes empresas. O código que serviu de base para o desenvolvimento deste projeto encontra-se disponível no *GitHub* (OPENNMT-PY, 2019), um ambiente de desenvolvimento de software aberto e colaboração (DABBISH *et al*.,2012).

Etapas de pré-processamento foram testadas com o uso de outros projetos, tais como o projeto Word2vec na conversão de palavras em vetores (MIKOLOV, 2013b) e as ferramentas da versão OpenNMT desenvolvidas na linguagem LUA (KLEIN, 2017).

2.5.1 HISTÓRICO DA VERSÃO OPENNMT-PY

A versão OpenNMT-py foi desenvolvida com o uso do PyTorch como biblioteca. O

PyTorch foi criado pela equipe de Inteligência Artificial do *Facebook* com a finalidade de demonstrar que era possível tornar o acesso à plataforma de software Torch compatível com a linguagem Python, o que resultou no PyTorch. O PyTorch é uma plataforma científica com suporte a aprendizagem de máquina de fácil paralelização. O que era para ser um exemplo, acabou se tornando uma potente estrutura de desenvolvimento, dada a facilidade de depuração e de acesso aos objetos para além de algoritmos padronizados.

O Pytorch, portanto, passou a funcionar não mais como uma biblioteca de software, mas uma plataforma de desenvolvimento de soluções com aprendizagem profunda compatível com Python. O PyTorch permite fazer uso de núcleos de processamento de GPU em lugar de CPU, o que será explicado com maiores detalhes nas subseções 2.5.2 e 2.5.3. Como resultado, a versão OpenNMT-py, que usa o PyTorch, é mais acessível para ajustes.

O projeto OpenNMT-py é compatível com o Tensorboard, um subproduto do Tensorflow. O Tensorflow foi desenvolvido pela Google e é a biblioteca usada na aprendizagem de máquina e na configuração de redes pela versão OpenNMT-tf. Os resultados apresentados pelos projetos OpenNMT-py e OpenNMT-tf são muito similares com vantagem. A versão que faz uso do PyTorch é mais flexível e melhor na paralelização de rotinas. A versão baseada apenas no Tensorflow geralmente é pouco mais rápida no desempenho geral. Alguns colaboradores contribuem com ambas as versões. A versão original OpenNMT, por outro lado, possui ferramentas mais detalhadas de pré-processamento de textos.

Existem outros motivos para a escolha da OpenNMT-py como base principal da solução, fruto da maior abertura dos recursos da parte do PyTorch, que são:

- pré-processamento de dados;
- processamento speech-to-text;
- processamento image-to-text;
- suporte à utilização de múltiplas GPU;
- modelos de camadas de rede do tipo RNN (LSTM / GRU) com *bridge*;
- possibilidade do uso de células convolucionais;
- modelo Transformer de tradução;
- monitoramento da aprendizagem da rede neural.

Conferências e workshops de máquinas de tradução automática (WMT), ocorrem mundialmente. Em mais de uma vez, tradutores a partir do projeto OpenNMT-py apresentaram bons resultados, com destaque em 2017 (KLEIN *et al*.,2017).

2.5.2 VETORES, MATRIZES E TENSORES

Vetores e matrizes são inerentes às redes neurais de tradução. Uma vez que cada palavra pode vir a se tornar um vetor, uma sequência de palavras corresponde a uma lista de vetores e pode ser armazenada em uma matriz de vetores. O mesmo pode-se dizer dos pesos de entrada e de cada camada escondida.

Existem bibliotecas matemáticas capazes de processar vetores e matrizes. Porém nem todas atendem aos cuidados que se fazem necessários:

- grande capacidade de representar os números e o seu arredondamento;
- alta velocidade de processamento multidimensional;
- memória de alto desempenho e grande capacidade de endereçamento;
- otimização e simplificação dos cálculos;
- processamento em paralelo.

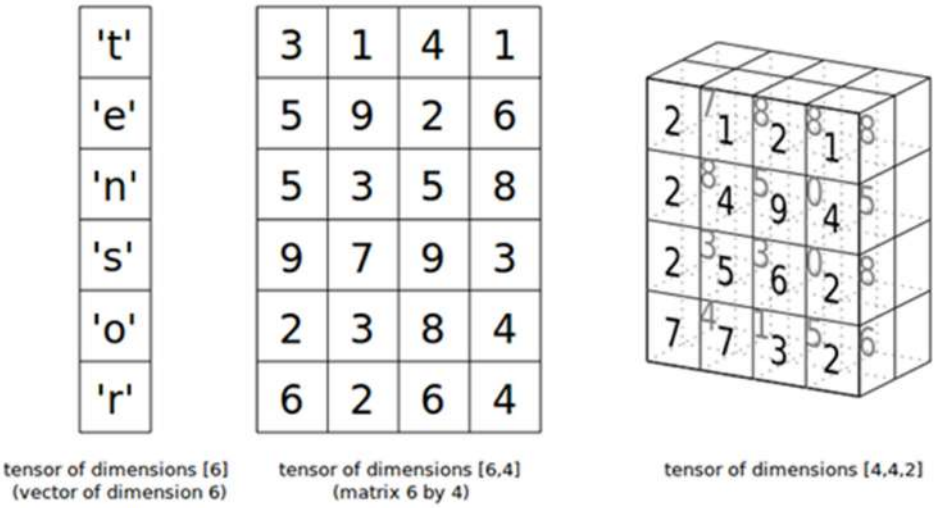
As redes convolucionais possuem filtros na forma de janelas que correspondem a matrizes, às quais são aplicados valores de entrada que também são matrizes. Por outro lado, redes recorrentes são preparadas para considerar os eventos anteriores no tempo. Como resultado, elas acabam formando matrizes de vetores e matrizes de matrizes em seus cálculos.

A representação de tais cálculos é complexa. O conceito e a representação de tensores foram criados para simplificar a visualização das equações, pelo que foi utilizado na Teoria da Relatividade. Porém, para fins deste relatório, basta considerar os tensores como matrizes de matrizes que possuem relações entre si (DANIELSON, 2018). Maiores detalhes podem ser encontrados no Apêndice A.

O primeiro exemplo à esquerda, na Figura 11, é um vetor de seis dimensões, ou uma matriz 1 x 6, ou um tensor de dimensão seis, em que cada linha contém uma letra do termo “tensor”.

No meio há uma matriz 6 por 4, ou um tensor de dimensões [6,4], e à direita há a representação de um tensor com as dimensões [4, 4, 2].

Figura 11- Representação de um tensor



Fonte: SQUARTINI (2013).

Ora, uma forma de se ver o último tensor é considerá-lo a partir das duas matrizes 4 por 4 que o formam. Se a primeira for chamada de T_1 e a segunda de T_2, então o vetor T de dimensão 2, ou ainda a matriz T de dimensão 1 x 2, em que a primeira coluna é T_1 e a segunda coluna é T_2, é um tensor $T = [T_1, T_2]$ equivalente ao tensor da Figura 11 mais à direita.

$$T_1 = \begin{bmatrix} 2 & 1 & 2 & 1 \\ 2 & 4 & 9 & 4 \\ 2 & 5 & 6 & 2 \\ 7 & 7 & 3 & 2 \end{bmatrix} \qquad T_2 = \begin{bmatrix} 7 & 8 & 8 & 8 \\ 8 & 5 & 0 & 5 \\ 3 & 3 & 0 & 8 \\ 4 & 1 & 5 & 6 \end{bmatrix}.$$

2.5.3 RECURSOS DE HARDWARE: CPU, GPU E MEMÓRIA

Nos primeiros ambientes criados para testes, fazendo-se uso apenas de CPU (*Central Processing Unit*), ficou clara a ineficiência do hardware para o treinamento do tradutor. Havia a necessidade de aquisição urgente de recursos com maior capacidade de processamento. Cem mil passos de treinamento levaram mais de 100 horas de execução a um consumo de cerca de 100% dos núcleos da CPU em praticamente todo o treinamento. Uma máquina virtual com oito núcleos e 16GB de memória RAM foi o máximo que se teve nos três primeiros meses.

Em ambiente virtual, o uso de CPU apresenta certas facilidades, tais como o gerenciamento e o custo de núcleos e memória. Contudo, o número de núcleos e a velocidade de transferência de dados das placas GPU (*Graphics Processing Unit*) são muito superiores. Porém, a memória da CPU não pôde ser aproveitada no processamento da GPU.

O laboratório contou com:

a) uma máquina virtual com 8 núcleos de CPU e 16 GB de memória compartilhada. Em dezembro a máquina passou a contar com 32 núcleos e 20GB de memória compartilhada;
b) dois computadores com CPU de 4 núcleos e uma placa GeForce com GPU GTX 1080 com 2560 núcleos e 8 GB exclusivos para o processamento da placa. O uso concomitante de duas GPU não foi configurado devido à falta de capacidade da fonte de alimentação do computador em manter as duas GPU e o espaço necessário para a refrigeração das placas.

Com o uso de GPU foi possível realizar até três testes de cem mil passos em um dia. Testes com maior volume de dados (mais de 3 milhões de linhas), que antes levavam uma semana, passaram a levar pouco mais de um dia. Em um dia, foram realizados vários testes menores a fim de analisar as respostas iniciais da rede com diferentes configurações, tais como número de entradas da rede, uso de camada bidirecional no codificador, número de dimensões dos vetores e taxa de aprendizagem.

O uso de GPU desviou a preocupação com a capacidade de processamento para a disponibilidade de memória. O consumo de recursos de memória é muito expressivo durante o treinamento (SENELLART *et al*., 2018). Recomenda-se verificar os tipos numéricos utilizados, pois isto influencia no consumo de memória. A diminuição na quantidade de memória necessária para os números pode ser obtida com a utilização de tipos numéricos que alocam menos memória (de float32 para float16, por exemplo).

2.6 METODOLOGIA

A metodologia adotada é própria de um relatório técnico-científico para elaboração de um produto e as seguintes pesquisas:

- Pesquisa exploratória – postura adotada desde o início do trabalho a fim de verificar a viabilidade do projeto e conhecer os modelos de tradução automática que apresentaram os melhores resultados em trabalhos acadêmicos e em competições.
- Pesquisa documental – necessária para a comparação e a avaliação da base teórica dos modelos encontrados na pesquisa exploratória, suas estruturas e seus métodos de otimização. Inclui também a releitura da evolução do referencial teórico do processamento de linguagem natural pertinente e os elementos cognitivos envolvidos nos aspectos linguísticos da tradução e da formação das palavras e das línguas.
- Pesquisa experimental – parte essencial que tomou bastante tempo da pesquisa (vide cronograma) por incluir a extração de dados, o desenvolvimento de novos programas, a adequação dos dados e os testes em laboratório. Inicialmente foram comparados diferentes modelos de redes neurais de tradução automática, que ao final limitou-se a dois tipos de redes neurais artificiais com suas variantes. Foram feitos ajustes em parâmetros que já faziam parte dos modelos e foram criados novos parâmetros. A pesquisa implicou o desenvolvimento e aproveitamento de programas para o tratamento prévio dos dados, na adaptação dos modelos às condições disponíveis na Câmara dos Deputados, no monitoramento do treinamento, na avaliação dos resultados e no serviço do protótipo em web.

2.6.1 PROCEDIMENTOS

Os procedimentos de laboratório seguem as seguintes etapas no processo de tradução:

1) seleção das fontes de dados;
2) coleta (ou extração) e armazenamento dos dados;

3) levantamento das arquiteturas e dos parâmetros;
4) correções e codificação necessárias ao aperfeiçoamento do sistema;
5) pré-processamento dos dados para treinamento;
6) monitoramento do treinamento e da validação com TensorBoard;
7) treinamento e validação;
8) avaliação dos resultados e melhorias;
9) testes com documentos sem tradução prévia;
10) especificação dos recursos possíveis de pesquisa.

As etapas foram repetidas conforme a necessidade apontada pelos resultados. Algumas vezes do início, outras vezes a partir de modelos intermediários que eram salvos durante o treinamento. Os aspectos analisados tiveram os seguintes pontos de atenção:

- seleção dos dados de treinamento para os fins propostos;
- tornar mais acessíveis as propriedades das palavras para a aprendizagem;
- não desconsiderar pequenos ganhos na qualidade da tradução;
- considerar a possibilidade de continuar o treinamento com parâmetros ajustados a fim de reforçar o treinamento.

2.6.2 SELEÇÃO DAS FONTES DE DADOS

A fim de que o tradutor seja especializado em documentos legislativos, a seleção das fontes de dados usou o conceito de sublinguagem (KUMAR; VENUGOPAL, 2017). Assim sendo, espera-se que as fontes de dados possuam estrutura, estilo, expressões e vocabulário próprios de documentos legislativos.

As fontes de dados devem ser provenientes de arquivos em formato de texto ou bases de dados com as sentenças. As sentenças podem ter a definição da língua, ou podem existir arquivos em paralelo, correspondentes linha a linha, à mesma sentença em determinado idioma (KOEHN, 2018). No caso de ser usado mais de um arquivo, deve-se cuidar para que os nomes identifiquem o documento e a língua, e por certo, o número de linhas deve ser o mesmo nos arquivos alinhados.

Os formatos sugeridos para a extração são:

- Páginas HTML contendo tabela com identificação da língua usada no documento, desde que haja paralelismo entre os documentos, por exemplo, uma coluna em uma língua e outra coluna, na mesma linha, em outra língua;
- Arquivos XML com marcação, ou identificação, da língua para cada sentença;
- Arquivos em Excel cujas colunas correspondam à mesma sentença na língua identificada na primeira linha na forma de cabeçalho;
- No caso de arquivos de texto do tipo CSV, as sentenças nas diferentes línguas devem estar em uma mesma linha e deve haver um cabeçalho na primeira linha a fim de identificar a respectiva língua por coluna;
- Respostas de serviços web específicos que respondam na forma de arquivos conforme os itens anteriores, ou ainda que seja possível identificar na resposta, para uma mesma sentença, pelo menos duas línguas distintas conforme solicitado ao respectivo serviço web.

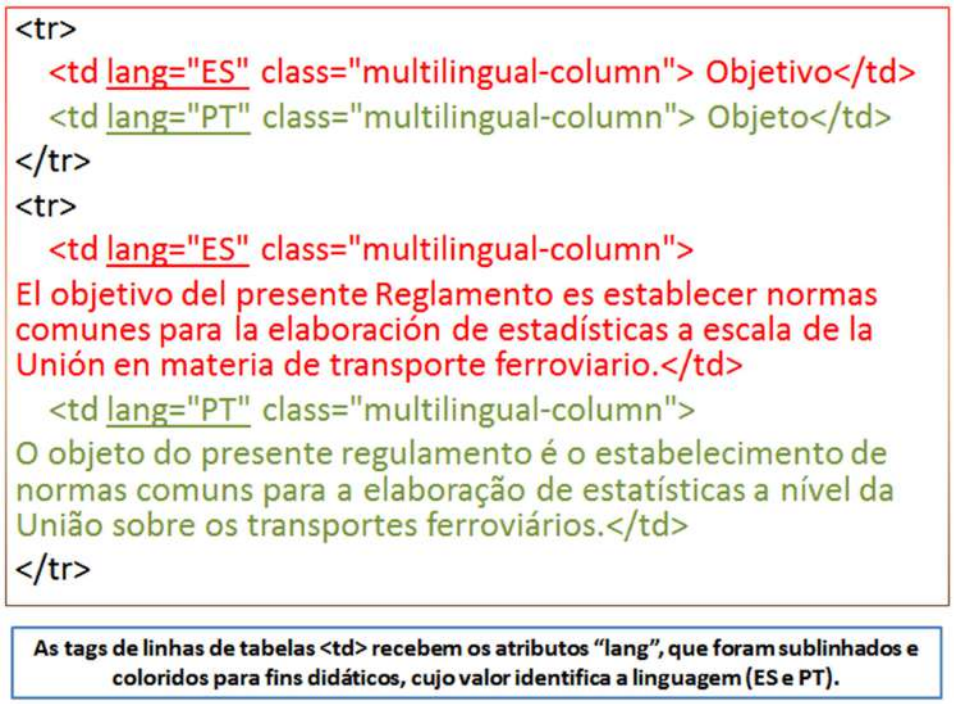
Uma estratégia adotada foi a de guardar, em base de dados, informações adicionais dos documentos extraídos, tais como a identificação da fonte de dados, do tipo de documento, data de publicação e localização, por exemplo. A identificação de elementos como título, capítulo, número do artigo, parágrafo ou item pode ser útil para lidar com a formatação. Neste caso, um programa irá salvar os arquivos em formato texto para uso no treinamento/validação e nos testes. Informações complementares podem ser extraídas de:

- serviços abertos de dados do Brasil ou exterior que respondam em mais de uma língua ou que retornem com a tradução de um texto enviado;
- sites internet que permitam a extração de dados em mais de uma língua desde que haja correspondência entre as linhas de um mesmo documento para cada língua;
- corpus específicos abertos já preparados;
- dicionários de domínio público em português europeu (PT-PT), português brasileiro (PT-BR), inglês britânico (EN-UK) e inglês americano (EN-US).

A Figura 12 mostra parte de um código HTML bilíngue extraído do EUR-LEX (UNIÃO EUROPEIA, 2018a). Os marcadores **<tr>** indicam o início de uma linha, a qual termina em **</tr>**. Cada coluna contém um idioma específico que é identificado pelo valor do parâmetro **lang**. Cada coluna começa com um marcador **<td>**, com os respectivos parâmetros, e termina com **</td>**. A primeira coluna de cada linha está em espanhol, enquanto que a segunda coluna da mesma linha está em português. A partir desse material pode-se fazer o

treinamento da tradução automática do Espanhol para o Português e vice-versa. Informações como o parâmetro **class** podem ser aproveitadas no momento de uma extração para identificação de tabelas HTML.

Figura 12- Exemplo de HTML com identificação da língua



Fonte: Elaboração própria (2018).

O formato de datas pode variar em uma mesma língua, como é o caso do inglês. Documentos formais em inglês britânico obedecem à sequência dia, mês e ano, com o nome do mês escrito por extenso, similar ao utilizado por países de língua portuguesa. O que significa que, apesar de estar em inglês o formato de data é diferente do adotado pelos Estados Unidos da América em documentos formais. O melhor é que o formato da data seja adequado ao contexto. No caso da tradução de um documento, manter o formato de data usado no país, bastando a tradução direta. No caso de ser um dado estruturado, usar formato único, independentemente do país de origem, a fim de facilitar a pesquisa.

Portais internet como o Euparl, do Parlamento Europeu, são ótimas fontes de dados para aprendizagem de máquina multilíngue contendo discursos. A partir do EUR-Lex (UNIÃO EUROPEIA, 2018a), é possível extrair leis em até três idiomas de uma vez. Conquanto tenha havido mudanças no EUR-Lex, o que exige uma atualização nos códigos utilizados nas extrações com Python, nada impede que as extrações sejam feitas em outras

linguagens de programação. A extração foi facilitada pela estrutura em nas páginas internet.

A mudança para uso de marcadores (*tags*) do tipo dificulta a extração nas páginas HTML. De toda forma, como os índices CELEX dos documentos de 2009 a 2017 foram guardados em tabelas, ainda é bastante simples extrair documentos alinhados com o Português e em quaisquer das línguas presentes no EUR-Lex.

Como fonte de dados para idiomas de países das regiões central e oriental da Europa, recomenda-se o projeto MULTEXT-East com documentos e arquivos multilíngues gratuitos (ERJAVEC, 2004).

As fontes de dados também podem incluir dados que já foram previamente tratados e disponibilizados na rede. Nesse caso, é possível encontrar arquivos para download contendo palavras no início de cada linha, as quais são acompanhadas de uma série de números. Tais números correspondem à representação vetorial da palavra ou termo correspondente. Os nomes dos arquivos, geralmente, indicam a língua e o número de dimensões dos vetores, conforme tratado na seção 2.3.4.

2.6.3 COLETA E ARMAZENAMENTO DOS DADOS

Com a coleta de dados seguindo os pressupostos já destacados para as fontes de dados, a extração a partir do EUR-Lex resultou em quase 80.000 (oitenta mil) documentos. Esses documentos, após o tratamento para a extração das sentenças, resultaram em mais de 8 milhões de linhas para cada uma das cinco línguas: Português, Inglês, Espanhol, Francês e Italiano. Estas línguas foram escolhidas pela proximidade entre elas, o que facilita a conversão do tradutor para um modelo multilíngue com compartilhamento de vocabulário.

No caso em questão, como foram armazenadas cópias em base de dados, cada linha de uma tabela pode conter uma coluna correspondente a uma língua. Apesar de uma linha poder conter mais de uma frase, a quebra da linha pode gerar erros de tradução. Antes do pré-processamento, recomenda-se eliminar linhas que apresentem incoerências (KOEHN, 2018). A qualidade dos textos irá afetar o treinamento e a qualidade da tradução final.

Dentre as ferramentas utilizadas n o manuseio dos dados estão os ambientes integrados de desenvolvimento Spyder (*Scientific Python Development Environment*) com a linguagem

de programação Python e o RStudio com a linguagem R. Em ambos os ambientes foram utilizadas bibliotecas científicas de software para a leitura e pré-processamento dos dados, ajustes de dimensionamento de escala, modelagem, treinamento, classificação, agrupamento e apresentação gráfica dos resultados. O uso de uma ferramenta ou outra dependeu da situação. Preferiu-se a linguagem R para fins de comparação de grande volume de dados, enquanto que o Python foi utilizado para a extração de informações de sites de dados abertos a partir da leitura de textos e tabelas em páginas web. Alguns dados puderam ser baixados diretamente da página Web. A organização dos dados e a filtragem antes dos treinamentos contaram com a ferramenta e base de dados SQLite/SqlBrowser no Linux.

Por conta da simplicidade da base de dados e do baixo desempenho com o uso apenas de CPU, preferiu-se guardar os dados em bases distintas de mesma estrutura e organizadas com base no ano. Assim, para cada ano foi criada uma tarefa no computador (*thread*), o que aumentou em cinco vezes a velocidade de captura e gravação de dados.

2.6.4 ARQUITETURA E PARAMETRIZAÇÃO

Os ajustes e as escolhas da arquitetura da rede neural são de grande relevância para a qualidade da tradução (BRITZ, 2017). Dentre os ajustes de configuração que mais se destacam estão:

- definir o número de dimensões dos vetores correspondentes às palavras a fim de que o computador possa processá-las;
- limitar, no pré-processamento, o tamanho máximo das sentenças e do vocabulário, qual a frequência mínima que uma palavra deve ter para entrar no vocabulário, e se ele será compartilhado ou não;
- ajustar o número de entradas da rede;
- número de camadas escondidas;
- tipo de rede neural artificial e otimização do gradiente;
- número de ciclos ou passos de treinamento;
- uso de vetor de contexto e agregação de posicionamento;
- tipo de vetorização, se por palavras ou por sentenças (*word embedding*);
- uso ou não de camada bidirecional na codificação;

- uso de ponte entre a codificação e a decodificação (*bridge*);
- inicialização dos pesos nas redes (principalmente no modelo Transformer);
- taxa de aprendizagem, decaimento e outros parâmetros correlacionados.

Alguns ajustes são inerentes à estrutura de rede neural adotada. Dentre os modelos que apresentaram melhores resultados com limitação de recursos, e que por isso foram os mais testados, foram aqueles que utilizaram a rede LSTM cuja primeira camada de codificação foi bidirecional. Conquanto essa rede seja bastante flexível em termos de ajustes, cuidados especiais devem ser tomados para se evitar a perda de sensibilidade da rede, ou ainda o efeito inverso, que pode ser um estado de saturação da rede, ou de grande oscilação, na aprendizagem.

O modelo de tradução Transformer também foi testado com bons resultados, mas é bastante restrito e sensível aos ajustes de parâmetros, o que dificultou a adequação do modelo aos recursos de máquina disponíveis. A limitação de recursos também limitou o uso de redes convolucionais, até porque, em se tratando de textos legislativos, é comum o uso de sentenças muito grandes.

Geralmente os modelos de redes neurais de tradução usam número de entradas na codificação igual ao de saídas na decodificação. No caso da normalização por termos (ou *tokens*) ao invés de sentenças, geralmente o número de entradas deve corresponder ao número de dimensões dos vetores que representam os termos.

Na prática, um sistema de tradução que faz uso de rede neural pode usar mais de um tipo de rede neural para cada módulo da rede. Redes neurais do tipo Feed-Forward, por exemplo, podem ser usadas para a definição de pontuação da probabilidade do uso de um determinado termo na tradução, uma vez que elas resultam no produto escalar entre a camada anterior e as entradas da próxima camada. Por outro lado, em havendo mais de uma saída, a função *Softmax* permite fazer a tomada de decisão quanto à melhor escolha.

Redes maiores são mais susceptíveis ao *overfitting*, situação em que a rede fica por demais ajustada aos dados do treinamento, mas se mostra ineficaz na tradução de novas sentenças. Uma das opções é o uso do recurso chamado de *dropout* (SRIVASTAVA *et al*., 2014), que significa desligar temporariamente as unidades (células ou neurônios artificiais) das camadas (visíveis ou invisíveis). O mesmo efeito pode ser obtido ao zerar temporariamente os pesos das conexões, ao invés de desligar as unidades. Originariamente a

escolha das conexões a serem desligadas é feita randomicamente. A redução no número de camadas também pode ser considerada como opção.

O fato é que o treinamento de cada modelo de rede neural tem suas peculiaridades, o que inclui a definição dos parâmetros definidos do treinamento, os quais também são chamados de hiperparâmetros. A presença da camada *multi-head attention*, por exemplo, é intrínseca ao modelo Transformer (VASWANI *et al*.,2017), assim como o uso da otimização de gradiente denominada Adam. Igualmente não faz sentido o uso de *dropout* quando a rede possui apenas uma camada escondida.

As escolhas das melhores configurações levaram em conta o desempenho da rede neural e o consumo de memória da GPU (SENELLART *et al*.,2018). Afetam diretamente o consumo de memória:

- o tamanho do vocabulário, algumas vezes chamado de dicionário;
- o tamanho dos vetores que representam palavras;
- o tamanho dos lotes no treinamento e na validação;
- o tamanho das sentenças.

No projeto OpenNMT é possível configurar a taxa de treinamentos e a taxa de decaimento da taxa de treinamento. Caso o decaimento leve a taxa de aprendizagem a valores muito baixos, pode-se: rever o método de otimização, definir um número de passos a partir de quando a rede deverá ser "reaquecida" (*warmup_steps*, em inglês), rever a taxa de aprendizagem, ou ainda reavaliar a qualidade do material usado no treinamento.

Para fins de aprendizagem, a utilização de tipos numéricos mais simples podem reduzir o custo operacional com as matrizes (SENELLART *et al*.,2018), o que leva ao consumo menor de memória e, geralmente, a um processamento mais rápido. Valores 0 e 1 podem ser representados por um bit ao invés de utilizar um tipo numérico de ponto flutuante. Contudo, tais otimizações não se encontram no escopo deste relatório.

2.6.5 AJUSTES ADICIONAIS NO PROJETO BASE

O projeto OpenNMT-py é um software livre, aberto à participação de quem nele esteja cadastrado. Isso significa que os ajustes realizados podem vir a fazer parte do projeto

principal. Porém, dentro do espaço aberto por aqueles que querem desenvolver algo a partir do projeto principal, há liberdade para que sejam mantidos os ajustes desejados sem interferir no projeto principal.

A preparação dos arquivos de entrada que se faz necessária para o treinamento da rede neural necessita de arquivos com dados específicos das línguas que serão usadas como entrada e como saída da rede. Entre esses arquivos há aqueles que evitam que abreviações ou palavras aparentemente isoladas seguidas de uma pontuação sejam separadas em termos distintos. Por exemplo: em “Sr. Paulo”, a frase não termina logo após “Sr”, pois “Sr.” é abreviatura do pronome de tratamento Senhor e não uma frase apenas com “Sr”. Por isso, tais arquivos são chamados de “nonbreaking_prefix”, cuja extensão do arquivo corresponde à língua a que ele se refere. Ocorre que o arquivo “nonbreaking_prefix.pt” não foi encontrado no projeto principal (a extensão “pt” correspondente à língua portuguesa). Como o mesmo tipo de recurso se faz necessário em máquinas de tradução estatística (KOEHN, 2018, p. 16), foi possível encontrar o referido arquivo no projeto MOSES, cujo código é aberto. Alguns ramos do projeto principal continham o referido arquivo, o que não justifica a sua ausência. A recomendação é um incentivo à pesquisa de tradutores com a Língua Portuguesa.

Outro ponto importante é a ordenação do projeto e atualização das informações de documentação que fazem uso de parâmetros em desuso e alguns erros ou melhorias já conhecidas que não foram ajustadas entre os módulos, que é o caso da tradução em mais de uma língua ao mesmo tempo.

2.6.6 PRÉ-PROCESSAMENTO DAS SENTENÇAS

A etapa de pré-processamento é a responsável pela preparação das palavras das sentenças antes do seu uso no treinamento e na tradução. Em geral, as redes neurais de tradução automática já fazem uso de um programa para o tratamento dos arquivos de texto que serão usados como entrada de dados. No caso da tradução, espera-se que haja um arquivo para a linguagem de entrada e outro para a linguagem de saída. Esses arquivos devem ser paralelos entre si no significado das sentenças.

O tipo de processamento mais comum é o da separação de palavras, números e outros sinais por meio da inclusão de um espaço. Também é possível segmentar as palavras e sinais por caractere e até compactar a segmentação fazendo-se uso de técnicas como BPE.

Assim, pares de caracteres próximos são convertidos em um caractere que não aparece no texto e deve ser guardada a correspondência com os pares de modo a permitir a decodificação. O funcionamento é bastante simples. A palavra “banana”, por exemplo, pode ser convertida em “baZZ”, caso em que a letra “Z” passa a corresponder ao par de caracteres “na”. O objetivo, nesse caso, é reduzir o tamanho da palavra. No caso da proposta de segmentação com BPE (SENNRICH; HADDOW; BIRCH, 2015), deve-se levar em conta a frequência com que a sequência de caracteres aparece no conjunto completo de palavras.

A partir da identificação de um erro de contagem dos termos do dicionário (vocabulário) no projeto básico do OpenNMT ficou clara a influência do tamanho do vocabulário no consumo de memória. Verificou-se, então, que o controle do vocabulário e da separação dos *tokens* pode servir de alicerce para que uma qualidade superior de tradução seja alcançada. Não se trata de uma mera redução no número de termos mantidos no vocabulário, mas de fazer com que as palavras consumam menos recursos sem perda de qualidade. A identificação das palavras é si é um recurso importante. Então, as alternativas propostas são:

1) que a estrutura da rede e as dimensões de vetores sejam tais que elas possam representar melhor as relações de associação e de construção das palavras;
2) que informações adicionais sejam incorporadas às palavras no vocabulário ou que as palavras sejam acompanhadas por informações que auxiliem o tradutor.

Palavras homônimas homógrafas trazem um problema para a representação vetorial de palavras. A solução quanto à presença de palavras homônimas homógrafas depende de “vetorização” prévia das palavras para fins de identificação do problema e de processamento posterior, o que não será tratado neste relatório, uma vez que a língua portuguesa apresenta poucas palavras homônimas quando se faz uso da sublinguagem do Legislativo.

Porém, alguns recursos foram adicionados a versões separadas dos arquivos “preprocess.py” e “inputter.py”, bem como aos programas de proteção e tokenização criados. Esses recursos foram criados com seguintes propósitos:

- normalizar, ou unificar, a representação na base UTF-8, especialmente para caracteres tais como as aspas duplas, separadores de propriedades (|) e outros que possam ser confundidos com caracteres especiais usados no tratamento do texto;
- proteger e identificar, tanto quanto possível, conjuntos de caracteres não

separados por espaço que não representam palavras, mas códigos, tais como endereços de e-mail e *hashtags*;

- separar pontuação ou agrupamentos que estejam concatenados com palavras;
- proteger espaços de formatação.

Conteúdos entre as tags <script> </script>, <?php ... ?> e outros do gênero, os quais não trazem necessariamente dados de formatação, assim como a conversão de documentos que não estejam em formato de texto (PDF, por exemplo) devem ser tratados antes do pré-processamento da rede.

2.6.7 MONITORAMENTO E TENSORBOARD

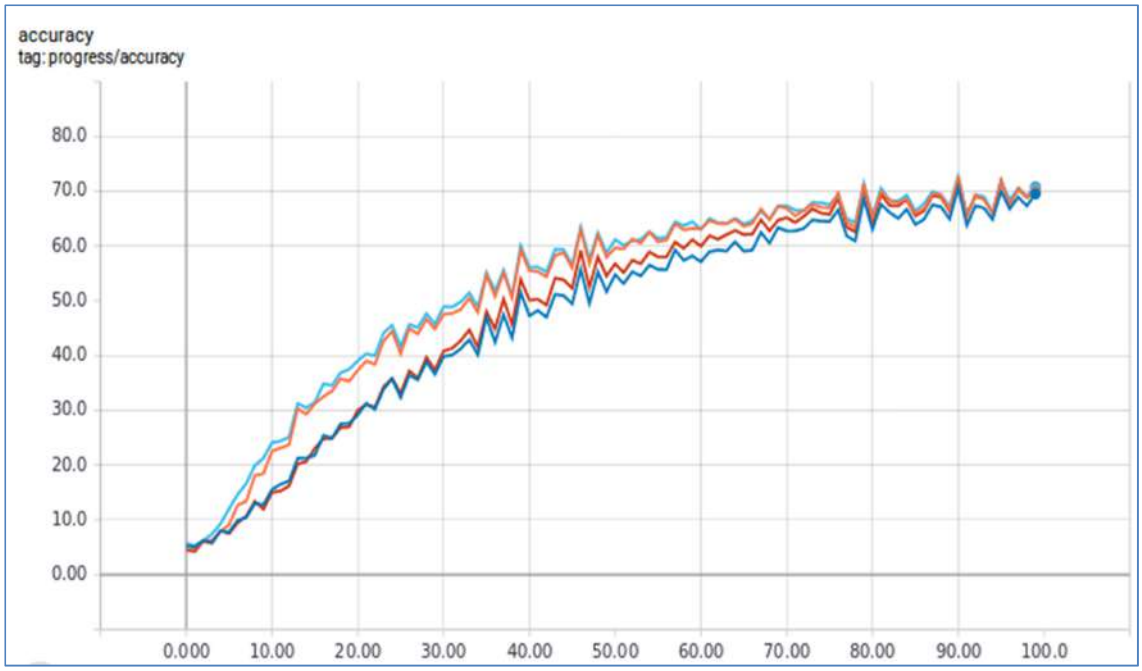
O monitoramento é muito importante durante o treinamento da rede, pois irá facilitar a análise da aprendizagem e pode até recomendar a interrupção precoce desta fase. Durante a execução do treinamento, são apresentadas algumas informações básicas sobre o andamento do treinamento, as quais são consideradas como indicadores de desempenho da rede (SENELLART *et al*., 2018).

O Tensorboard é um recurso que permite o acompanhamento visual do treinamento por meio de gráficos e permite a comparação entre treinamentos. Também é possível comparar o progresso a partir dos dados do treinamento e dos dados de validação. Os registros (*logs*) alimentam os gráficos no Tensorboard, e podem ser baixados a partir da página web do Tensorboard. Conquanto o Tensorboard acompanhe o Tensorflow da Google, ele pode ser utilizado por outras implementações, bastando que os dados sejam gravados em formato compatível e seja apontado o diretório que contém os dados.

A Figura 13 apresenta um dos gráficos do Tensorboard usado para a comparação entre quatro treinamentos distintos realizados com os mesmos dados. As pequenas mudanças nos parâmetros de treinamento incluíram ajustes no número de entradas, o número de camadas da rede e conexões adicionais. Todos os treinamentos cumpriram o número de 5 mil passos definido no início do treinamento . No eixo das ordenadas está o progresso da acurácia (*accuracy*), cujo cálculo se faz dividindo o número de palavras traduzidas corretamente (leia-se, igual à sequência de saída esperada) pelo número total de palavras. Ela é apresentada na forma de percentual.

A Figura 13 mostra também a importância de se fixar um valor de semente (*seed*) para facilitar a comparação dos treinamentos. Vê-se que alguns movimentos das linhas parecem sincronizados, pelo que, apesar das configurações serem distintas, os treinamentos consumiram tempo similar. Se o objetivo é uma aprendizagem mais rápida, a configuração representada em azul parece melhor. Porém, importa ainda observar que, apesar da linha vermelha crescer mais lentamente, ela acaba ultrapassando as demais no final.

Figura 13- Comparação entre 4 treinamentos



Fonte: Elaboração própria (2018).

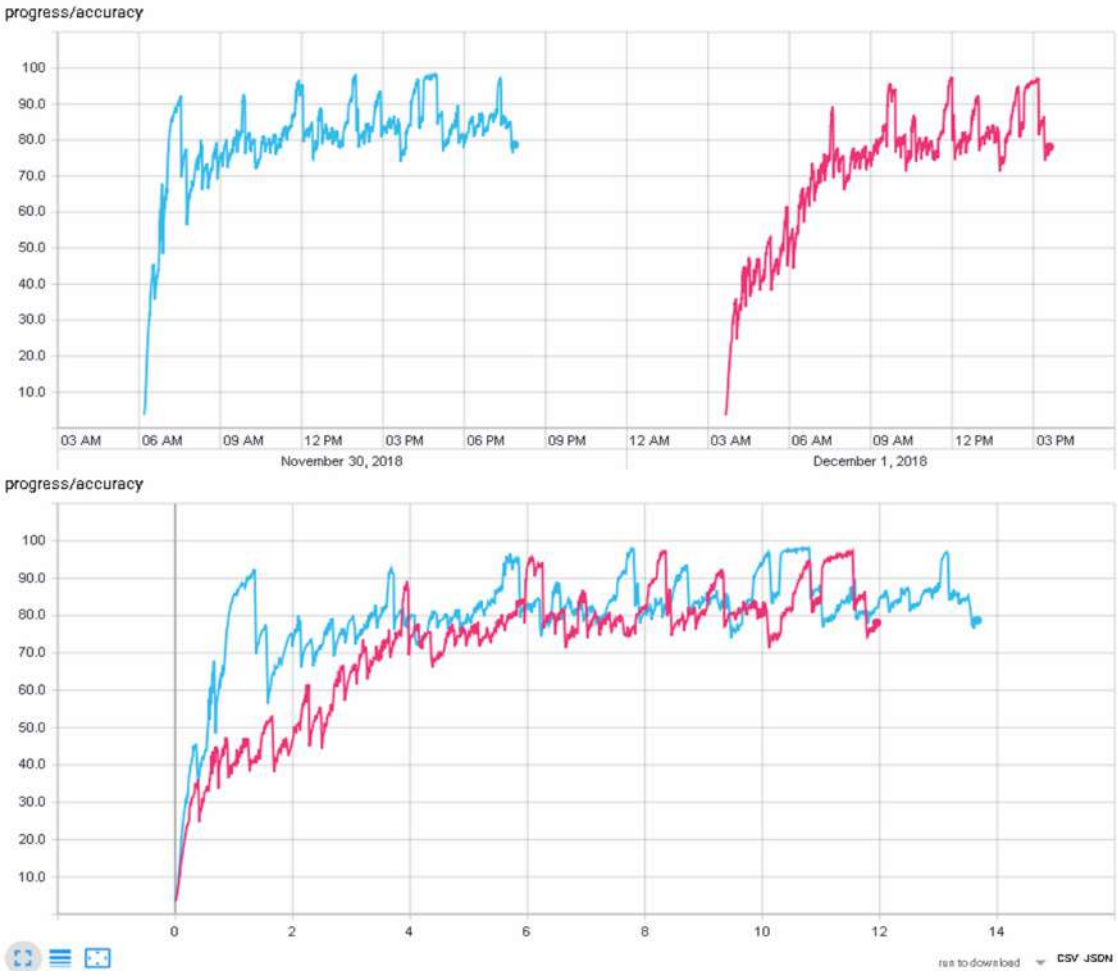
Dos poucos casos em que houve expressiva queda na acurácia durante o treinamento, seguida de estagnação, caracterizando a deterioração do gradiente, a maioria foi no modelo Transformer. Isso pode ser justificado pela dificuldade de se usar a estrutura recomendada, seja quanto ao número de seis camadas, seja quanto à memória reservada ao processamento em lote, sem que houvesse estouro de memória. Assim sendo, o número de testes com redes recorrentes foi muito maior do que com modelos convolucionais ou Transformer.

2.6.8 TREINAMENTO

O aprendizado das redes neurais utilizado é do tipo supervisionado, isto é, existe uma resposta certa que deverá ser aprendida pela rede neural. Os dados são divididos em

treinamento, validação e teste. De pouco vale ter um resultado excelente no treinamento, se o resultado da validação se mostrar muito aquém do desejado. Os testes feitos com a utilização de dados bilíngues permitem a avalição automática, sem tradutor humano. O treinamento fez uso de dados previamente processados nas duas línguas selecionadas. A língua portuguesa foi considerada como entrada (*source*) e a inglesa como saída (*target*). Levando-se em conta o grande número de sentenças obtidas, inicialmente foram separados 80% para treinamento e 20% para avaliação dos resultados, proporção considerada suficiente em caso de grande quantidade de dados. Recomenda-se a utilização de um valor de semente fixa a fim de permitir a comparação entre os treinamentos de um mesmo tipo de rede. Nesse caso recomenda-se ainda que sejam alterados no máximo dois parâmetros na estrutura da rede a fim de viabilizar a efetiva comparação entre os treinamentos. Caso se queira ver os efeitos na mudança dos dados de entrada, recomenda-se fixar os parâmetros da estrutura da rede.

Figura 14- Diferentes formas de visualização



Fonte: Elaboração própria (2018).

A Figura 14 ilustra dois gráficos com os mesmos dados com duas formas diferentes de visualização baseadas no tempo do treinamento, WALL (gráfico superior) e RELATIVE (gráfico inferior). O primeiro gráfico mostra o treinamento em ordem cronológica, enquanto o segundo mostra os dois treinamentos iniciando juntos. O treinamento em vermelho levou 12 horas, enquanto o em azul levou quase 14 horas.

Em suma, a abscissa dos gráficos de monitoramento dos indicadores de treinamento do Tensorboard pode apresentar os valores a partir das seguintes opções:

1) STEP – indica o número de passos divididos pelo intervalo indicado no parâmetro *report_every* do treinamento. O número de passos real é o resultado da multiplicação do valor apresentado pelo valor atribuído ao parâmetro;
2) RELATIVE – compara o tempo de treinamento em horas alinhando o seu início;
3) WALL – mostra o tempo absoluto em horas em uma linha de tempo.

2.6.9 VALIDAÇÃO

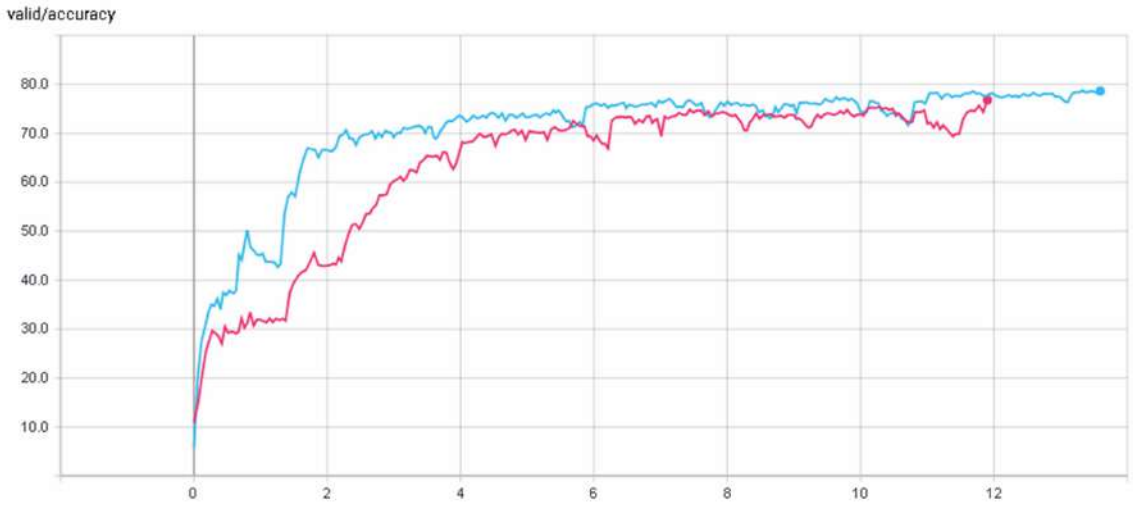
A validação integra o acompanhamento do treinamento no Tensorboard e é esperado que a melhoria nos resultados durante o treinamento seja acompanhada pela melhoria na validação. É natural que os resultados do treinamento sejam melhores do que os da validação, mas se não houver uma melhoria proporcional isso está a indicar *overfitting* na rede, isto é, o modelo está preparado apenas para os dados para os quais foi treinado. Nesse caso, pode-se até diminuir as dimensões da rede, adicionar mais dados ao treinamento, embaralhar a sequência dos dados ou ainda aumentar o valor do *dropout*.

Determinados ajustes em hiperparâmetros implicam na obrigação de reiniciar o treinamento desde o início, sendo possível manter os estados da rede e dar continuidade ao treinamento. No caso de se retornar ao treinamento após uma interrupção, deve-se verificar a pasta onde foram gravados os relatórios para o Tensorboard a fim de garantir a continuidade do monitoramento. Se o número total de passos foi alcançado, a continuidade dependerá de aumento no número total de passos do treinamento. Mudanças nos dados de treinamento devem garantir o vocabulário já treinado. Não se deve abrir mão da validação como indicador no treinamento.

A Figura 15 apresenta resultados de uma validação dos treinamentos da Figura 14.

Caso a abscissa estivesse na apresentação no modo STEP, como os intervalos entre as validações geralmente é diferente, o mesmo ocorreria com os valores da abscissa. Como o modo de apresentação foi o mesmo utilizado no segundo gráfico da Figura 14, RELATIVE, o acompanhamento ficou mais fácil. As principais diferenças foram a atenuação na oscilação da validação e a menor acurácia (*accuracy*), o que é natural, uma vez que os intervalos entre as validações foi maior e os dados de validação são distintos do treinamento.

Figura 15- Validação durante o treinamento



Fonte: Elaboração própria (2018).

Outro indicador relevante para a evolução do treinamento da rede é o da perplexidade (*ppl*), que indica a capacidade de predição do modelo. Quanto menor a perplexidade maior a capacidade preditiva do modelo. Então, no início do treinamento, o valor é muito grande. Porém, geralmente após os primeiros mil passos, a visualização em escala logarítmica é preferível no acompanhamento da perplexidade durante o treinamento.

O cálculo da perplexidade é feito a partir da Equação (2), em que N é o número de termos da sentença e $P(\mathrm{w_1 w_2 w_3 \ldots w_N})$ é a probabilidade de que as palavras ocorram em uma dada sequência.

$$ppl = \sqrt[N]{\frac{1}{P(\mathrm{w_1 w_2 w_3 \ldots w_N})}} \qquad (2)$$

Então, quanto maior a probabilidade de acerto, menor o valor da perplexidade. Quanto maior a perplexidade, mais difícil para o tradutor tomar a decisão certa. Naturalmente, a tendência é de que, quanto maior a acurácia dos resultados, menor seja a

perplexidade. No início do treinamento é esperada uma grande variação no valor da perplexidade, o que é atenuado no decorrer do treinamento. A convergência desejada é a aproximação dos resultados em relação ao texto já traduzido usado como referência. Também é natural que ocorram oscilações durante a aprendizagem e que estas diminuam com o tempo. A redução na taxa de aprendizagem leva em consideração a convergência do treinamento ou pode ser definida em número de passos.

2.6.10 AVALIAÇÃO DA TRADUÇÃO

Os testes de tradução são realizados com parte da fonte de dados não utilizada. A maior fonte de dados bilíngues ou multilíngues encontrada foi o EUR-Lex. Ocorre que, em termos de ortografia, o Novo Acordo Ortográfico deixou uma ampla liberdade quanto ao uso de acentos no Português de Portugal e do Brasil. Por exemplo, COMITÉ (PT-PT) e COMITÊ (PT-BR) possuem o mesmo sentido e ambas as grafias são aceitas.

As avaliações humanas de traduções feitas por máquinas possuem custo elevado e podem levar meses de trabalho humano que dificilmente poderá ser reutilizado (PAPINENI *et al*.,2002). A comparação dos resultados por humanos e por máquinas é bem diferente. Avaliadores humanos de tradução levam em conta:

- a adequação da tradução em transmitir o significado original do texto, sem distorcer, perder ou adicionar informação;
- a fluência na tradução envolve tanto a correção gramatical como a escolha das palavras.

Enquanto isso, os modelos de avaliação da tradução automatizada partem de uma ou mais referências do que seria a tradução correta. A equipe de pesquisa de máquinas de tradução da IBM propôs um método automático de avaliação denominado BLEU (*Bilingual Evaluation Understudy*) com o objetivo de automatizar a própria avaliação da tradução. A intenção era a de minimizar a necessidade de intervenção humana na tradução e facilitar o trabalho do profissional de tradução em caso de necessidade. Por isso, o método valoriza a ordem das palavras em relação ao que seria uma tradução humana. Na avaliação, leva-se em conta a distância entre as palavras e aplica-se uma penalidade se o tamanho da sentença é menor do que o correspondente à melhor tradução (PAPINENI *et al*., 2002). A avaliação,

portanto, irá considerar o número de combinações possíveis, os pesos das associações dos termos ou a proximidade dos termos. O resultado destas combinações deve variar de 0 a 1.

O uso do processo de avaliação se tornou uma referência na busca pelo estado da arte em tradução automática (COLLOBERT *et al*., 2011). As características desejadas de uma técnica de avaliação automática da tradução são:

a) rapidez e baixo custo no processo de avaliação;
b) independência em relação às linguagens envolvidas;
c) proximidade com a avaliação humana.

Porém, o fato é que pode existir mais de uma tradução correta (REITER, 2018). Uma avaliação mais precisa poderia fazer uso de aspectos léxicos e semânticos combinados com a estrutura gramatical. Uma fonte de consulta é a rede BabelNet, que reúne informações léxicas e semânticas das palavras (NAVIGLI; PONZETTO, 2012). A definição do campo léxico tem a ver com a pertinência de uma palavra a uma área de conhecimento, que organizando a palavra dentro de seu contexto, inclui as formas de composição e de derivação da mesma. O campo semântico de uma palavra, por sua vez, abrange o emprego e os significados que ela pode ter (FERREIRA, 2009; HASSAN *et al*., 2018). A viabilidade do uso de uma determinada palavra poderia ser feita através da comparação com estruturas gramaticais válidas. De certo modo, o tradutor aprende o uso das palavras em tais estruturas, o que justifica o fato de que o problema das variantes não é apenas ortográfico, mas também no modo de formar as sentenças.

2.6.11 DE VOLTA AO TREINAMENTO

A maior parte do tempo foi ocupada com ajustes de hiperparâmetros e tratamento de dados a fim de conseguir os melhores resultados praticamente no limite da memória. Os melhores resultados, onde os *tokens* corresponderam a palavras, foram obtidos com redes LSTM, codificador bidirecional, ponte entre decodificador e codificador, vetores de palavras com dimensões entre 400 e 600, e redes com 500 a 800 entradas.

No caso de utilização de caracteres, o número de entradas aumenta, pois cada letra se torna em uma representação, pelo que o número de entradas chegou até mil entradas. No caso de uso de caracteres, o uso de codificador bidirecional é menos relevante, mas é

imprescindível o uso de conexões residuais e *dropout*, especialmente na utilização de variantes de uma mesma língua, como foi o caso do Português do Brasil e de Portugal. O modelo torna-se mais resiliente a alterações ortográficas.

Quando os resultados são promissores, mas carecem de pequenos ajustes, recomenda-se a interrupção do treinamento e sua retomada após os devidos ajustes. Sempre se lembrando de garantir a continuidade do monitoramento informando a pasta onde estava sendo feita a gravação dos indicadores e revendo o número máximo de passos. É recomendável rever a taxa inicial de aprendizagem e de decaimento, a menos que se trate do modelo Transformer, cuja taxa de aprendizagem inicial é fixa. Alguns hiperparâmetros não podem ser modificados.

2.6.12 DESAFIOS E OBSTÁCULOS

Dentre os principais desafios, destaca-se o baixo volume de dados para treinamento com Português do Brasil. A limitação do vocabulário é feita com base na frequência das palavras no conjunto total de textos. A limitação por frequência pode ser explícita ou associada ao número de palavras definido para o vocabulário. A conversão ou substituição de palavras dos documentos extraídos do EUR-Lex, do Português europeu para o brasileiro, apenas atenuou o problema. O maior volume de dados com Português europeu fazia com que algumas palavras usadas com maior frequência no Brasil do que em Portugal ficassem ausentes no vocabulário.

Contudo, as diferenças vão além da ortografia. Conforme já fora abordado consoante aos termos específicos ligados ao poder legislativo de cada país, é de grande relevância o uso de glossários ou bases de termos legislativos. O glossário de termos legislativos do Congresso Nacional começou a ser traduzido mais ao final da pesquisa, de modo que não houve tempo hábil para adaptar e testar o uso no treinamento e no processamento de textos para a tradução automática.

Aproveitar as melhorias das atualizações do projeto OpenNMT-py, usado como base do tradutor, sem perder o que estava sendo construído, também foi um grande desafio e apresentou alguns obstáculos. O maior obstáculo foi a quebra de compatibilidade no que diz respeito ao uso de propriedades agregadas às palavras, ou *tokens*, pelo projeto OpenNMT-py, um importante recurso na redução do tamanho do vocabulário e na utilização da segmentação

por caracteres. Tais propriedades, desde o projeto original OpenNMT, eram agregadas e concatenadas às palavras durante a tradução. Apesar do recurso continuar disponível no treinamento e ter apresentado ótimos resultados, ele passou a gerar um erro. O fato piorou ainda mais com a migração para o PyTorch 1.0, cuja primeira versão foi muito instável.

Após identificar que o erro no tratamento das propriedades estava associado às dimensões dos tensores e que ele sempre ocorria ao final de uma sentença, constatou-se que a solução não seria simples. Migrações e atualizações do projeto haviam perdido a compatibilidade com o tratamento de *tokens* especiais, o que explicava o vácuo no tratamento de início e fim de sentença no servidor de tradução. A correção não seria pontual e poderia ainda afetar diretamente as rotinas de otimização, o que somado às novas formas de processamento de grandes volumes de textos, recurso criado próximo à migração para o PyTorch versão 1.0, tomaria o disponível para os ajustes e testes finais para a apresentação do produto à União Interparlamentar. O caminho alternativo encontrado foi o de substituir a concatenação das propriedades pelo uso de *tokens* de marcação específicos, os quais antecederiam as palavras. Eles seriam incluídos após a proteção no processo de tokenização para que o processamento após a saída da rede.

O maior obstáculo para o início dos treinamentos foi a aquisição da placa com GPU. Por ser muitas vezes confundida com uma simples placa de vídeo, a comparação de preços com produtos muito inferiores exigiu muito esforço para esclarecer os requisitos técnicos. A aquisição também enfrentou a questão do valor limite para aquisição, pelo que foi possível a compra de até duas placas para desktop dentro dos preços encontrados. Placas para computadores servidores (*workstations*) são muito mais caras do que as placas para computadores de uso pessoal, tipo desktop, ainda que, por vezes, elas possam ser melhores. A avaliação das placas levou em conta a pontuação geral, a velocidade de processamento de operações e o preço.

Uma vez adquiridas as placas, verificou-se que os computadores desktop poderiam suportar apenas uma placa, visto que duas placas GTX 1080 exigem uma fonte com pelo menos 800 watts de potência. Por outro lado, nos servidores cujas fontes de alimentação eram capazes de manter as placas, a posição dos conectores PCI-Express 3.0 era tal que a saída de ventilação das placas ficaria abafada. Desktops do tipo torre, cujas fontes de alimentação eram mais potentes, não possuíam conectores PCI-Express 3.0, apenas PCI-Express 2.0. Ainda que compatíveis em barramento, a velocidade da PCI-Express 2.0 é menor do que a da PCI-

Express 3.0. Ao final, a alternativa foi manter dois desktops do tipo torre, cada um com uma placa, em conector PCI-Express 2.0.

O uso de uma fonte externa não foi descartado, porém, com o início do funcionamento do protótipo para a apresentação, verificou-se que o tempo de resposta do serviço de tradução web era muito alto, razão pela qual um dos computadores passou a ficar dedicado ao serviço web.

Apesar de trazer grande ganho na velocidade do treinamento e na tradução, o uso dos núcleos da GPU preconiza o uso da memória da GPU. Quanto maior a rede, maior o consumo de memória. Uma solução seria encontrar um computador que suportasse ambas as placas, visto que o PyTorch é compatível com o uso de mais de uma placa GPU no tradutor.

Dentre as principais estratégias para se vencer o obstáculo da restrição de memória, sem aquisição, estão: segmentação de palavras; redução nas dimensões da rede; redução nas dimensões dos vetores que representam as palavras; redução do número de palavras do vocabulário; redução no número de sentenças consideradas por otimização do treinamento; redução no número de sentenças consideradas por otimização na validação. A segmentação de palavras, especialmente a segmentação por caracteres, traz economia significativa de memória em termos do vocabulário, mas também implica em aumento no tamanho das sentenças por token, o que exige que a entrada da rede neural artificial seja maior. Bibliotecas como a Fastext agregam vantagens associadas à estrutura das palavras, mas o tamanho do dicionário ainda continuará o mesmo.

3. RESULTADOS

É importante lembrar que a aprendizagem adotada é a supervisionada, isto é, as respostas são avaliadas em termos de acerto ou erro. A resposta serve para realimentar a rede com o objetivo de minimizar os erros e tais informações estão disponíveis no monitoramento.

Ainda sem a instalação da placa de alto desempenho com GPU, foram feitas diversas simulações que contaram com um número menor de passos de treinamento e um espaço menor entre os intervalos de validação da rede neural a fim de analisar os efeitos da arquitetura da rede. Dos melhores resultados encontrados usando LSTM simples com duas camadas, duas arquiteturas se mostraram muito similares mudando apenas a velocidade de processamento, ambas com palavras com no máximo 500 dimensões.

A otimização padrão por meio do gradiente descendente estocástico foi mantida para as redes LSTM por resultarem em melhores curvas de decaimento da perplexidade e entropia cruzada, o que condiz com os resultados de outros pesquisadores (KLEIN *et al*., 2017). A utilização de rede bidirecional na codificação mostrou melhores resultados, mesmo que isso implique na redução pela metade do número de entradas da primeira camada.

O uso de uma ponte entre a última camada do decodificador (*decoder*) para a primeira camada do codificador (*encoder*) aumenta o paralelismo, compartilhando informações da saída para a entrada, o que colabora com o cuidado com a disposição e escolha dos termos (WU *et al*., 2016). Tal configuração obteve resultados positivos na aceleração da redução da perplexidade no início dos treinamentos, mas apresentou instabilidade no final do treinamento quando o número de camadas escondidas da rede neural foi de apenas uma. Outra característica do uso de apenas uma camada escondida da rede neural é que não é possível fazer uso de um recurso chamado de dropout, o qual pode ser aplicado em redes do tipo LTSM. O objetivo de, em determinados momentos do treinamento, eliminar temporariamente algumas das ligações da rede, é o de evitar que esta não fique tão adaptada aos dados de treinamento ao ponto de funcionar bem apenas com eles. Esse efeito é denominado de *overfitting*. Evitá-lo é tornar a rede mais robusta à variação de cenários.

Os indicadores principais por ordem de maior relevância foram: acurácia (*accuracy*), a perplexidade (*ppl*) e a entropia cruzada (*xent*). A perplexidade sinaliza melhor a resposta inicial de ajuste da rede. A entropia cruzada, que corresponde à função de custo dividida pelo número de palavras apresenta melhor visualização no decorrer do treinamento.

3.1. EVOLUÇÃO E COMPARAÇÃO

O primeiro teste realizado alcançou um valor BLUE compatível com outros testes apresentados em trabalhos acadêmicos e apresentou uma taxa de até 93% de acurácia na aprendizagem. Apesar do bom resultado, ainda não havia sido feita a otimização da arquitetura para a tradução PT-EN e o volume de dados era menor devido ao tempo gasto pelo processamento unicamente pelos núcleos da CPU (até 4 dias corridos). Mesmo assim o resultado foi bom, comparando-se com valores publicados na internet.

Como já foi dito, um indicador utilizado para avaliar a tradução é o BLEU. Os valores alcançados nos primeiros testes foram decisivos para se considerar o uso do projeto OpenNMT:

Quadro 3- Avaliação da Tradução

Bilingual Evaluation Understudy - BLEU			
24,54 56,7 / 32,1 / 20,1 / 13,4			
BP	**ratio**	**hyp_len**	**ref_len**
0,928	0,930	22340	24013

Fonte: Elaboração própria (2019).

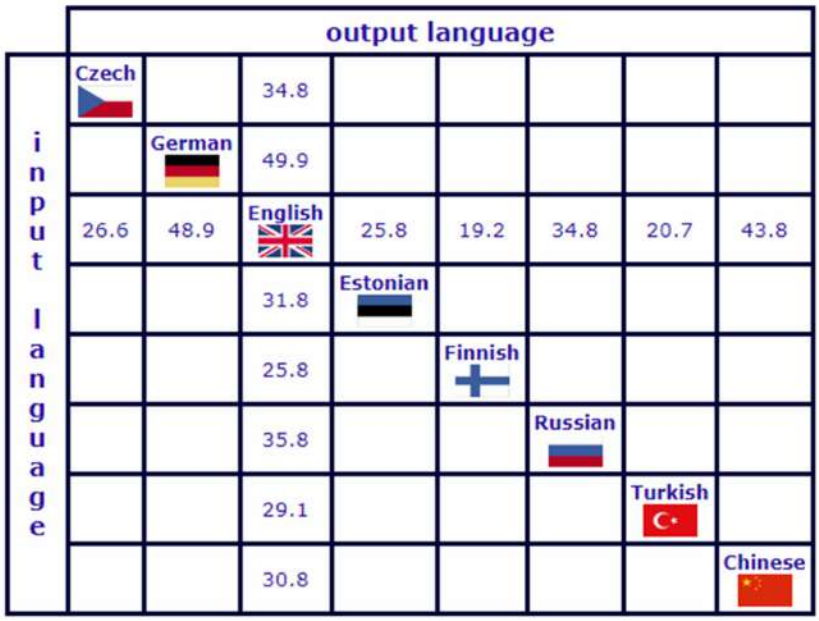
Para se entender os parâmetros é importante considerar que a comparação se baseia em *n-grams*, isto é, a qualidade da tradução leva em consideração a ordem da sequência das palavras. O valor de *hyp_len* corresponde somatório dos tamanhos considerado da hipótese, que corresponde ao resultado esperado da tradução, enquanto que *ref_len* corresponde ao somatório dos tamanhos da tradução da referência. A *ratio* é o resultado da divisão de *hyp_len* por *ref_len* (*hyp_len* / *ref_len* =22340 / 24013). O valor de BP corresponde ao valor da penalidade aplicada às sentenças menores (*brevity penalty*) do *corpus* completo. O primeiro valor, BLEU, é o valor mais considerado como resultado do teste. Ele é acompanhado de uma sequência de números que correspondem aos valores para cada n-gram. Como são quatro valores, ele pode ser chamado de BLEU4. O primeiro valor, 56.7, é o valor considerando uma palavra e o número máximo de vezes que uma palavra aparece.

A utilização da GPU acelerou o processo de treinamento e permitiu aumentar o número de passos nos treinamentos. A inclusão da GPU também trouxe novas exigências, tais como a necessidade de definição do nível de informações a serem apresentadas e se o número de placas a serem consideradas na otimização, uma vez que pode-se fazer uso de mais de uma placa GPU. Assim sendo, as duas placas poderiam participar dos ajustes do gradiente durante

o treinamento.

A utilização do Tensorboard foi importantíssima para facilitar a comparação entre os resultados na medida em que eram feitos ajustes. Os gráficos comparativos, a partir dos respectivos logs de treinamento, permitiu descartar os modelos que apresentaram piores resultados e aprimorar os demais. Os gráficos também serviram de alerta de problemas com as fontes de dados e de necessidade de ajustes na rede em treinamento.

Figura 16- Melhores resultados alcançados em 2018



Fonte: EuroMatrix (2019).

A Figura 16 apresenta uma matriz com os melhores resultados de tradução obtidos por sistemas de tradução automática com a técnica de comparação BLEU. *Input language* corresponde à língua de entrada no tradutor, e *output language* corresponde à língua de saída do tradutor. Maiores detalhes sobre os sistemas utilizados podem ser encontrados no endereço internet do projeto EuroMatrix (2019).

É natural que, ao se falar do desenvolvimento de um tradutor, seja levantada a questão sobre a viabilidade de se usar tradutores automáticos de empresas tais como a Google, a Microsoft e outras. A avaliação passa pelos preços dos serviços, as demandas e a qualidade do serviço.

O Quadro 4 apresenta, à esquerda, recursos do projeto da Google no WMT 2016, e à direita, recursos do Laboratório para o protótipo. Ambos são tradutores baseados em redes neurais artificiais. Em verde destacam-se as vantagens, enquanto em vermelho as

desvantagens. O WMT 2016 foi um workshop de tradução automática em que a Google se destacou. O Laboratório do tradutor também apresentou resultados muito bons e possui estrutura semelhante ao trabalho apresentado pela Google em 2016. Com respeito à viabilidade de uma tradução automática, cuja qualidade seja suficiente para uso na tradução de todas as leis do país, ou pelo menos de um número considerável dessas leis, fez-se a comparação do protótipo com os recursos de redes neurais da Google. Na ocasião, os resultados da Google, considerados os recursos disponíveis, foram considerados como o estado da arte em tradução automática.

Quadro 4- Comparação entre Google e Lab CD

	Google NMT (WMT 2016)	CD NMT Lab (2018)
Tipo de RNA	LSTM	LSTM
Camadas escondidas	8 (1 bidirecional)	2 ou 3 (1 bidirecional)
Unid. Processamento	GPU	GPU
Modelo GPU	Tesla k80 (2 x k40)	GTX 1080
Memória por GPU	24GB	8GB
Número de GPU	8	1
Fontes de dados	Big Data, Genérico	Legislativo, Sublinguagem
PCIe (Slot)	3.0 (3.0)	3.0 (2.0)

Fonte: Elaboração própria (2019).

Sobre as comparações quanto aos recursos de hardware, a placa GTX 1080 usada no Laboratório tem melhor desempenho que uma placa Tesla k80. Porém, a quantidade de memória da Tesla k80 é bastante superior, além do fato do conector do computador utilizado no protótipo ser PCI-Express 2.0 não permitir o uso da placa GTX 1080 em sua capacidade máxima. Pode-se dizer que houve equilíbrio quanto ao processamento das placas, porém a Google utilizou uma placa por camada, isto é, oito vezes mais recursos de processamento e vinte e quatro vezes mais memória.

Verificou-se a necessidade de ampliação dos recursos de memória a fim de obter maior acurácia na tradução para a pesquisa interparlamentar. Por outro lado, os resultados confirmaram a tese de que tradutores especializados em uma sublinguagem apresentariam maior qualidade na tradução, como foi o caso do protótipo, do que se fazendo uso de grande volume de dados não especializados. O protótipo, portanto, alcançou resultados satisfatórios e muito promissores, além de confirmar a viabilidade da tradução de leis e o acerto na forma de se restringir o vocabulário à sublinguagem de um grupo social específico: o Poder Legislativo. Ressalta-se que os primeiros testes com o uso de GPU obtiveram resultados com indicador BLEU equiparáveis aos melhores resultados de eventos acadêmicos de tradução

automática com aprendizagem de máquina.

3.2. APROVEITAMENTO DOS VETORES DE PALAVRAS

Os arquivos com vetores de palavras, assim como o dicionário dinâmico, podem ser extraídos do treinamento após a preparação dos dados. Estes recursos estarão disponíveis e poderão ser utilizados para tradução e para outras necessidades em processamento da linguagem natural (NLP, *Natural Language Processing*). Os vetores de palavras podem ser extraídos dos modelos de tradução e terão as dimensões utilizadas no treinamento. Quando gerados no treinamento, eles irão consideram a partir do modelo baseado em sequência,

A Câmara dos Deputados, por sua vez, está implementando o projeto Ulysses, que, diferente da página de pesquisa da nuvem, irá contar com um *chatbot*, uma pequena interface automatizada, geralmente semelhante à imagem de uma pessoa ou robô, que procura criar um ambiente mais informal de busca de informações. Por ela o usuário do portal internet da Câmara dos Deputados poderá tirar dúvidas e ser encaminhado para as páginas mais relevantes conforme a necessidade por ele expressa. Tanto o *chatbot*, como a sumarização e outras pesquisas poderão fazer uso desses recursos linguísticos. A sugestão levada à União Interparlamentar é que haja a disponibilização de tais recursos no âmbito da UIP.

3.3. PROTÓTIPO DE PÁGINA E SERVIÇO WEB

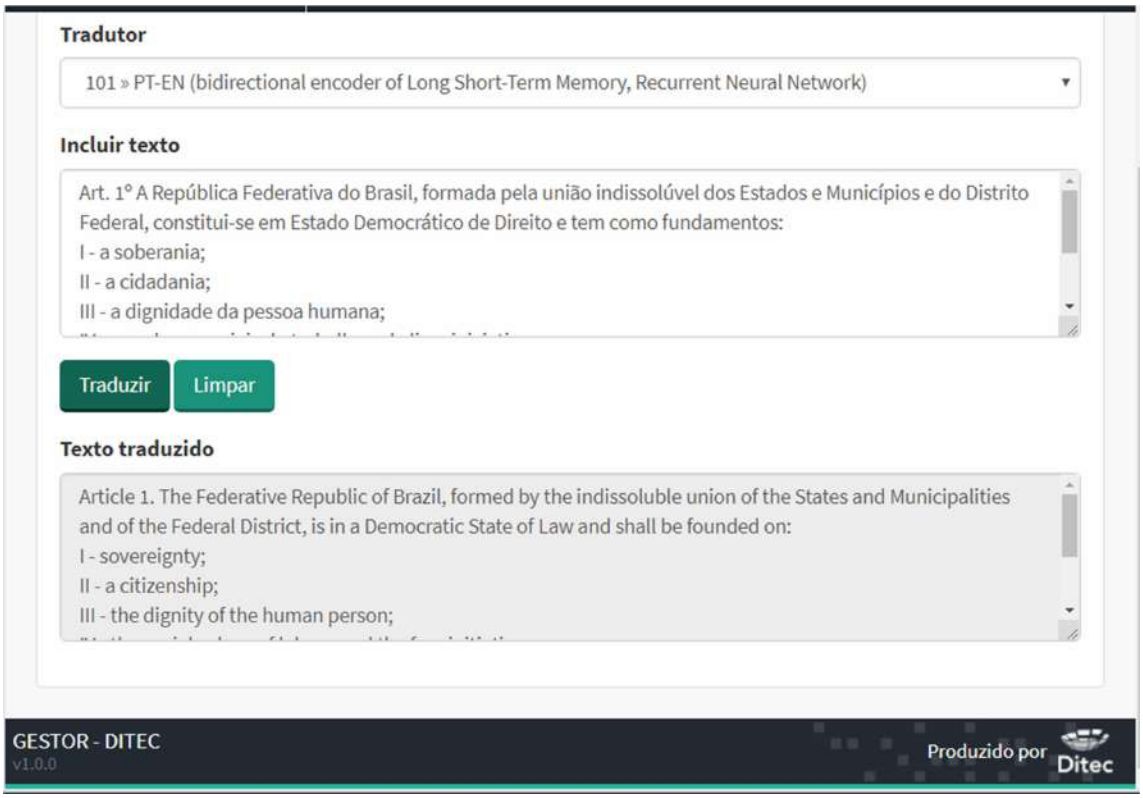
A primeira aplicação disponível na página protótipo do Ulysses foi o aproveitamento e adaptação da página desenvolvida como protótipo pela equipe do Sistema Eletrônico de Votação. A página segue os padrões da Câmara dos Deputados e acessa um serviço Web em *Flask* que carrega o tradutor.

A página foi apresentada em sua versão em inglês. Estiveram presentes representantes da Câmara dos Deputados, Senado Federal, Parlamento Europeu, Parlamento Pan-Africano, Canadá, Chile, Finlândia, Estônia, Espanha, Ucrânia, Israel e África do Sul.

A Figura 17 mostra a parte principal da página web que pertence ao protótipo apresentado na reunião com a União Interparlamentar na Câmara dos Deputados. O serviço inicialmente era executado no computador com na máquina virtual com 32 núcleos de CPU,

sem GPU. O mesmo computador que era usado no pré-processamento de vetores de palavras. O tempo de resposta do serviço de tradução, em alguns casos, ultrapassava 20 segundos. Por isso o serviço foi migrado para um dos computadores com GPU. O resultado foi que a tradução, em geral, não ultrapassava cinco segundos.

Figura 17- Visão parcial da página web do protótipo do Tradutor Automático



Fonte: Elaboração própria (2019).

O seletor abaixo do título "Tradutor" foi criado para poder mostrar mais de uma opção, a com o uso da rede recorrente, LSTM com codificador bidirecional, e com o uso do Transformer, que acabou sendo desativado. Preenchido o campo do texto a ser traduzido, basta pressionar o botão de traduzir para receber a tradução. Como descrito no Apêndice A, dentre as correções feitas, foi tratado o erro no caso de envio de linha em branco e no uso do parâmetro "name" que estava sendo ignorado pelo código original do projeto OpenNMT-py.

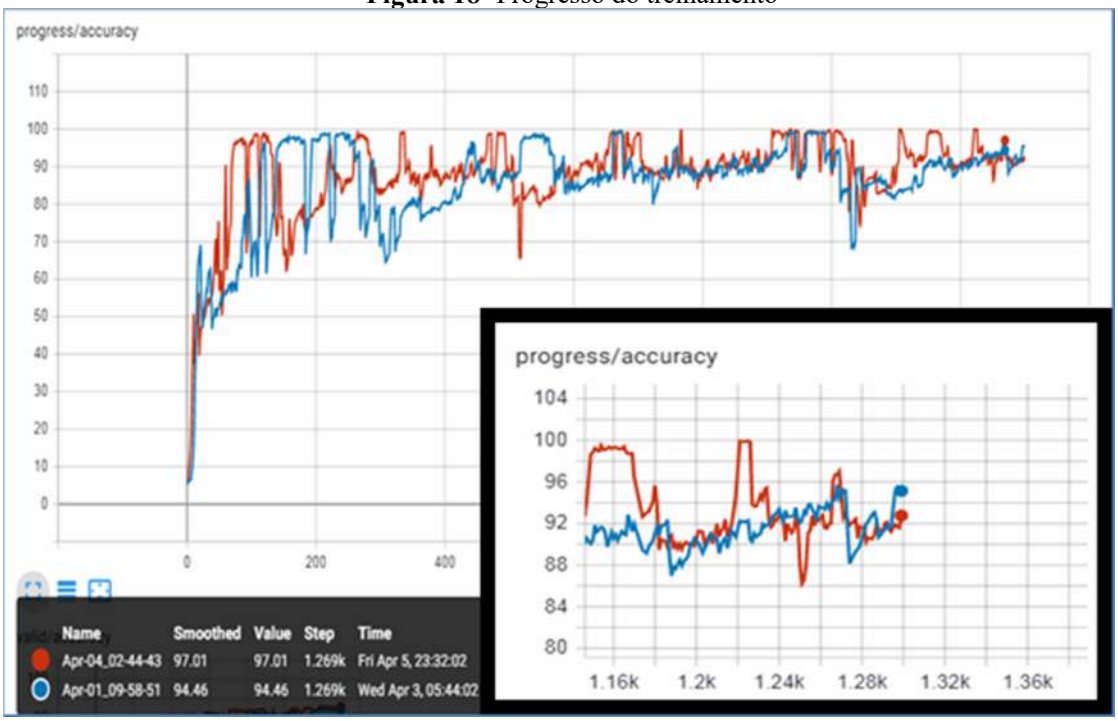
Também houve mudança no funcionamento do serviço web, que aparentemente foi criado considerando o processamento de várias sentenças ao mesmo tempo, mas que na realidade, no código então presente na versão principal do projeto, considerava tudo como uma única linha. Por isso, inicialmente havia uma grande preocupação com o tamanho máximo da sentença, uma vez que o processamento era feito em CPU.

Ao procurar melhorar a velocidade da tradução, percebeu-se que algumas estruturas previstas do serviço eram subutilizadas. As alterações tornaram-se uma oportunidade para que a tokenização e a tradução mantivesse estruturas de dados, incluindo os tensores, mais compatíveis. Ao final, o serviço web passou a fazer a tradução com a tokenização, linha a linha, com bom desempenho.

3.4. PROGRESSOS DOS TREINAMENTOS E VALIDAÇÃO

Fazendo-se uso apenas de documentos legislativos da União Europeia extraídos do EUR-Lex, chegou-se a alcançar, com o tratamento dos textos, uma média de 96% de acurácia no treinamento e o máximo de 92% de acurácia na validação, lembrando que a validação usa textos distintos dos usados no treinamento. O cálculo da acurácia pode ser consultado no Apêndice C.

Figura 18- Progresso do treinamento



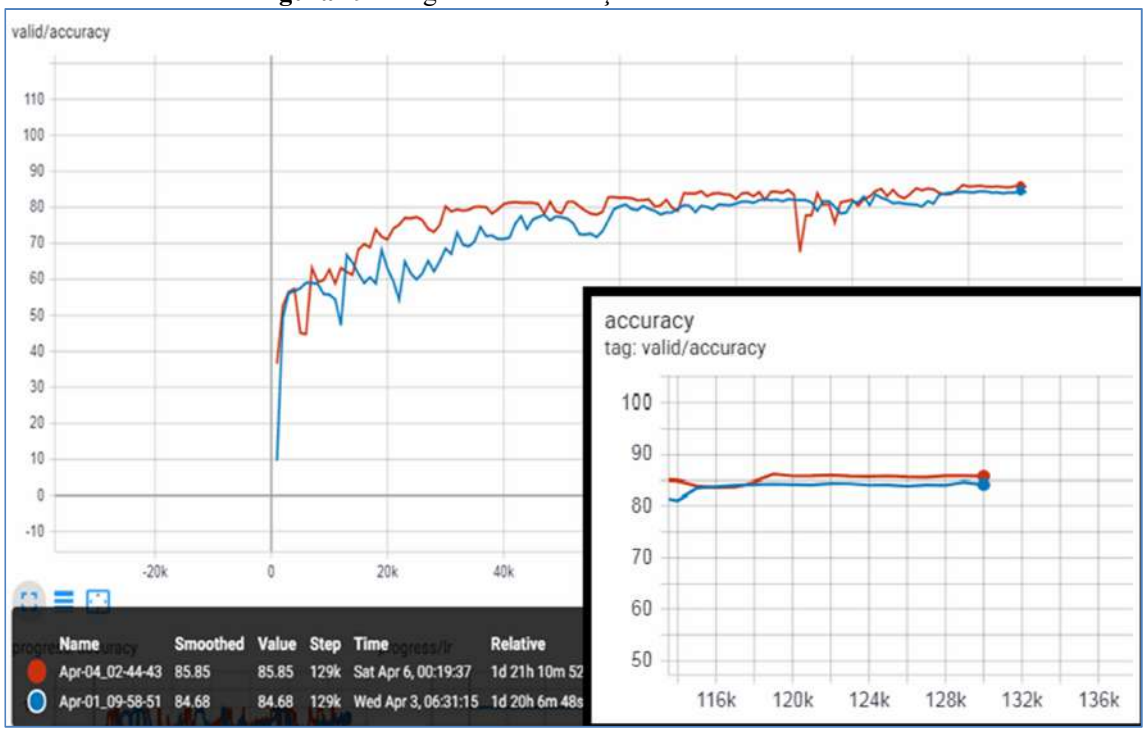
Fonte: Elaboração própria (2019).

A Figura 18 mostra o resultado de dois treinamentos em que foram incluídas leis brasileiras aos textos aos textos que já vinham sendo utilizados. Mesmo com a conversão de

palavras para a grafia utilizada no Brasil e com a fragmentação das palavras em caracteres, a inclusão da variante brasileira da língua portuguesa levou à queda no desempenho em relação aos resultados anteriores. O recorte dentro da Figura 18 representa o final de dois treinamentos ampliados para melhor visualização da escala. O valor médio da acurácia nos últimos passos foi de 92%. A instabilidade apresentada nos treinamentos é resultado da redução do número de lotes de sentenças, também chamados de *minibatches*, utilizados para o cálculo médio dos erros e ajuste do gradiente, por motivo de limitação de memória.

Houve predominância da variante linguística do Português europeu, tanto no treinamento como na validação. Foi necessário compensar a falta de textos bilíngues com a variante do Português usado no Brasil para garantir a frequência mínima necessária para a inclusão no vocabulário. Mais de cem palavras foram migradas da segundo as regras do Acordo Ortográfico da ortografia predominante na Europa para a predominante no Brasil. Os resultados, apesar de promissores, apresentaram queda tanto no treinamento como na validação, o que pode ser constatado na Figura 19.

Figura 19- Progresso da validação durante o treinamento

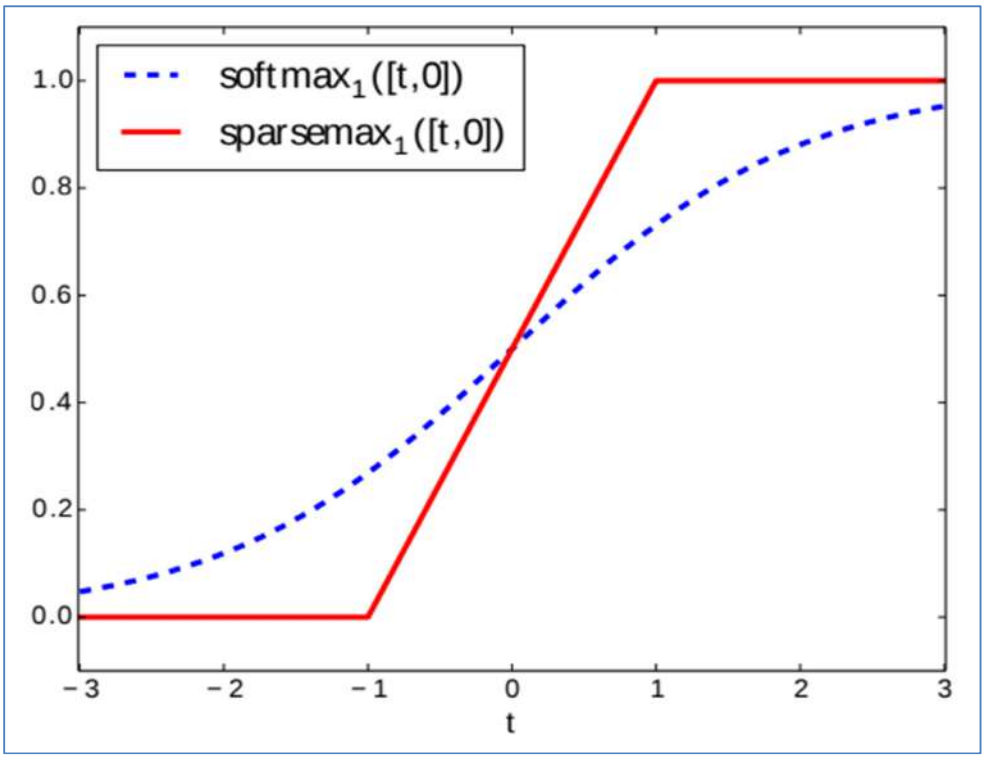


Fonte: Elaboração própria (2019).

A menor instabilidade na validação é resultado do uso de intervalos maiores do que os usados no treinamento. Nos testes os primeiros passos de treinamento poderiam ser

suficientes para a interrupção do teste. Vários testes foram interrompidos antes mesmo de chegarem a 60% de acurácia. Quando um treinamento se mostrava muito similar a um teste anterior, mas com resultados inferiores, geralmente era logo interrompido. A depender do tipo de ajuste, especialmente no caso de parâmetros associados à otimização da aprendizagem (algoritmo de gradiente, função de custo, *dropout*, conexões, taxa de aprendizagem, etc) ou ao preparo dos textos, um maior número de passos se fazia necessário para a avaliação. Um treinamento mais demorado não implica necessariamente em um final com pior resultado.

Figura 20- Funções Softmax e Sparsemax



Fonte: Martins e Astudillo (2016).

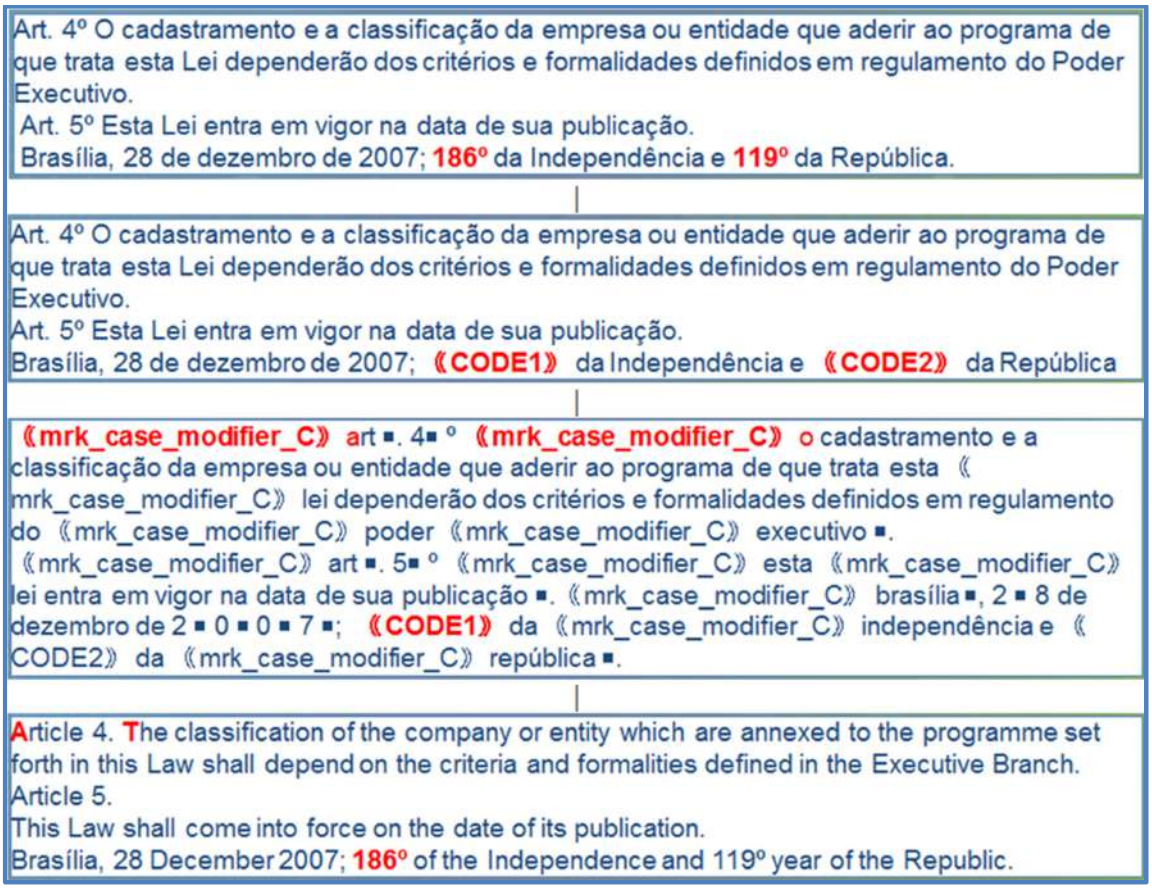
A diferença entre treinamentos similares geralmente se deve a pequenas diferenças nas dimensões ou tamanho de rede, na inicialização dos pesos e na escolha das funções a serem utilizadas. Tal foi o caso da preferência pelo uso da função *Sparsemax* em detrimento da *Softmax* (MARTINS; ASTUDILLO, 2016), apenas para fins de avaliação das probabilidades na predição de saída a partir do vocabulário usado. Conforme a Figura 20, a função *Sparsemax* se parece com a *Softmax*, porém, por resultar em valores iguais a zero ou 1, em determinada faixa de entrada, a *Sparsemax* é menos susceptível a ruídos do que a *Softmax*. O Apêndice C contém os cálculos dos principais indicadores usados e as funções *Softmax* e

Sparsemax podem ser combinadas no cálculo da função de perda ou custo. As mesmas funções podem ser usadas na camada de atenção.

3.5. FLUXO DA TRADUÇÃO DE DOCUMENTOS

A Figura 21 mostra os principais passos na tradução automática de um trecho da Lei nº 11. 637/2007. Na primeira caixa de texto, de cima para baixo, são apresentados os artigos 4º e 5º da lei. Caixas de texto apresentam em sequência os efeitos das principais operações descritas neste relatório no processo de tradução. Foram destacados em vermelho alguns dos elementos do texto que sofreram mudanças.

Figura 21- Passos na tradução automática



Fonte: Elaboração própria (2019).

A segunda caixa de texto da Figura 21 apresenta o resultado da proteção de dois termos com números ordinais. A terceira caixa de texto mostra o resultado de uma tokenização com a identificação do primeiro caractere em maiúsculo, de modo que a

tokenização não interfere na proteção de termos. Logo após a tokenização, todas as palavras são escritas em minúsculo (exceto os códigos protegidos), pelo que cabe às marcações a identificação da mudança para maiúsculo. No caso de mais de uma letra da palavra estar originariamente em maiúsculo, um tipo específico de token é utilizado para identificar a região em maiúsculo. O texto, então, seguirá para a entrada do tradutor automático.

Para chegar à tradução final em Inglês, o texto passa pela operação inversa da tokenização e, a depender dos códigos utilizados, segue para a recuperação dos termos previamente protegidos e para a substituição direta de palavras ou expressões com tradução previamente definida. Enfim, o resultado da tradução do texto na versão em Inglês.

3.6. LIÇÕES APRENDIDAS

O desenvolvimento, com recursos da inteligência artificial, de uma ferramenta de tradução automática treinada a partir de documentos do contexto do Poder Legislativo, a fim de facilitar a pesquisa a documentos legislativos em mais de uma língua, esteve sempre associado ao projeto da Nuvem Interparlamentar. As lições aprendidas integram este relatório e têm por objetivo transmitir parte da experiência adquirida na execução do projeto com o objetivo de subsidiar trabalhos futuros (PMI, 2009). As lições aprendidas que mais se destacaram foram as seguintes:

- Atenção à quantidade total de memória para o processamento em GPU disponível, pois ela poderá passar a ser o principal limitador dos recursos.
- O uso de mistura de variantes de uma língua no treinamento deve ser feito com cautela. Na situação de carência de dados, considerar a conversão de palavras para a variante que será usada na tradução, tomar medidas que aumentem a frequência de palavras utilizadas na variante linguística pretendida e, se for o caso, identificar no início da linha a variante utilizada.
- Evitar o uso de CPU no treinamento e no ambiente de produção (conquanto neste caso a demanda de memória seja menor). Se os recursos forem poucos, usar o computador com mais CPU e sem GPU para o pré-processamento de dados. O uso de GPU para treinamento e tradução pode acelerar o processamento em cerca de 10 vezes.

- Incluir glossário e base de dados de terminologias na tradução como fontes de dados de treinamento e recursos para a tradução direta. A recomendação dada aos parlamentos foi que eles não se limitem a apenas o seu idioma e o Inglês. Recursos que devem estar disponíveis a outros parlamentos.
- Reconhecer a importância do trabalho dos tradutores humanos. A língua é um sistema bastante complexo e dinâmico. A Inteligência Artificial foi desenvolvida a partir do que foi encontrado na natureza, como quase tudo em Ciências. Ainda que o objetivo da tradução seja a assimilação da essência de um documento, recomenda-se a contratação de tradutores humanos, os quais podem auxiliar, tanto no *feedback* dos erros mais grosseiros encontrados na tradução automática, como também podem colaborar com a tradução das páginas de conteúdo na internet.
- O pré-processamento dos textos antes do treinamento tem relevante impacto no aprimoramento do tradutor automático. Aspectos linguísticos não devem ser descartados, uma vez que eles enriquecem a tradução. Recursos que viabilizem um maior volume de informações sobre as sentenças e as palavras colaboram com o tradutor e, consequentemente, com o resultado da tradução.
- Não descartar o processamento em paralelo de mais de um modelo de tradutor no processo de tradução para fins de comparação. A qualidade na tradução de sentenças curtas ou longas varia de acordo com o modelo de rede neural artificial e com o processo de tokenização adotados.
- A tradução automática não é justificativa para a ausência de tradutores humanos. Tradutores humanos têm papel relevante na melhoria da tradução automática na produção de novas fontes de dados multilíngues, na verificação de erros do tradutor automático e na manutenção da base de termos de tradução direta e do glossário.

4. CONCLUSÃO

A viabilidade do projeto foi confirmada pelos resultados apresentados na primeira reunião do *hub* da Nuvem Interparlamentar de Dados Abertos, o que inclui o protótipo que integrou o projeto Ulysses, da Câmara dos Deputados, sob a responsabilidade da Diretoria de Inovação e Tecnologia da Informação. Resultados que foram alcançados com menos recursos de hardware se comparados aos eventos de workshop de tradução automática.

Foram identificados os requisitos necessários ao desenvolvimento de um tradutor automático, de modo que países, ou grupos de países, que desejem desenvolver um tradutor especializado em determinada área de atuação ou determinados tipos de documentos, podem usar este relatório como base. As características próprias da língua não devem ser ignoradas e um trabalho cooperativo deve ser incentivado como forma de aproximar os povos.

Pequenas melhorias na qualidade da tradução não devem ignoradas e o aproveitamento de técnicas antigas no tratamento de textos não deve ser descartado. Uma vez que tradutor automático teve de funcionar, ao mesmo tempo, no modo de tradução por documento e no modo de serviço internet online, o tratamento prévio dos textos teve de ser tornar mais robusto. O tratamento dos textos unificado incorporou técnicas usadas na versão original OpenNMT, técnicas da versão em Python e o desenvolvimento de programas que unem filtragem e proteção de termos.

Em se tratando de recursos de tradução, é essencial a comunicação ecooperação entre países. As mudanças realizadas no projeto OpenNMT-py estão compartilhadas no *GitHub* e apresentadas em reunião realizada no Brasil, cuja apresenta é encontrada no Apêndice D. Além de programas, é importante também que sejam compartilhadas bases de dados de terminologias e glossários de termos legislativos de todos os países que disponibilizam seus dados a fim de sanar dúvidas e colaborar com a tradução. Subprodutos da tradução podem ser utilizados para a tradução humana e também devem ser disponibilizados. Arquivos de vocabulário que incluem o número de ocorrência das palavras são recursos úteis para a avaliação de aplicações de processamento da linguagem natural (NLP).

É importante enfatizar que a tradução automática não tem valor legal, não se trata de tradução oficial juramentada, mas recurso de apoio à pesquisa interparlamentar. Seu uso deve auxiliar a identificar e pesquisar os pontos principais de que trata uma lei, conhecer o modo como uma determinada questão foi tratada por um país, conhecer diferentes perspectivas e

abordagens sobre um tema e conhecer mais sobre a cultura de outras nações.

A tradução automática pode ser essencial a países de elevada produção legislativa que não têm, por obrigação, que publicar suas leis em outras línguas com igual valor legal. Outros países podem utilizá-la como apoio à tradução humana. Países membros da União Europeia, que possuem serviço de tradução automática disponível, a princípio não precisam se preocupar com a implementação de semelhante produto. No caso de países como o Brasil e a Ucrânia, por exemplo, com maior volume de produção de documentos legislativos, a tradução automática tem papel muito mais relevante.

Recomendam-se seguintes opções para eliminar o problema de restrição de memória:

a) aquisição de novas placas com memória a partir de 24 GB, para o caso de uso de estação de trabalho servidora;
b) aquisição de computador que suporte as duas placas GTX 1080 com PCI-Express 3.0, tal que a montagem não comprometa a refrigeração das placas, sabendo-se que cada placa usa espaço correspondente a dois conectores;
c) contratação de serviço de nuvem com GPU que atenda aos treinamentos, lembrando que a tradução demanda menos memória do que o treinamento.

Como projetos futuros podem ser citados o aperfeiçoamento da tradução automática por meio de abordagem que leve em consideração a complexidade da linguagem e sua construção e dinâmica. O término da tradução do glossário de termos foi previsto para o mês de agosto de 2019. Ele permitirá por em prática a proteção e tradução direta de termos legislativos. É esperada também a ampliação do Laboratório de modo a permitir que os serviços de tradução automática sejam fornecidos em ambiente de produção.

REFERÊNCIAS

AGRAWAL, Prakhar. Question Generation for Reading Comprehension.**IIT Delhi**, 2017. Disponível em: http://www.cse.iitd.ac.in/ques_gen/report.pdf. Acesso em: 4 jan. 2019.

BA, Jimmy L.; KIROS, Jamie R.; HINTON, Geoffrey E. **Layer Normalization**. 2016. Disponível em: https://arxiv.org/pdf/1607.06450.pdf. Acesso em: 5 dez. 2019.

BAHDANAU, Dzmitry; CHO, Kyunghyun; BENGIO, Yoshua. **Neural machine translation by jointly learning to align and translate**. 2014. Disponível em: https://arxiv.org/pdf/1409.0473. Acesso em: 4 nov. 2018.

BOOTH, Andrew D. Calculating machines and mechanical translation. **Discovery**, v. 15, n. 7, p. 280-285, 1954. Disponível em: http://www.mt-archive.info/Discovery-1954-Booth.pdf. Acesso em: 4 nov. 2018.

BASSETT, D. S. *et al*. Task-Based Core-Periphery Organization of Human Brain Dynamics. **PLoS Computational Biology**, [S. l.], v. 9, n. 9, p. 1–16, 2013. Disponível em: https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=90530168&lang=pt-br&site=eds-live&scope=site . Acesso em: 1 jun. 2019.

BRASIL. **Constituição da República Federativa do Brasil**: texto constitucional promulgado em 5 de outubro de 1988, com as alterações determinadas pelas Emendas Constitucionais de Revisão no 1 a 6/94, pelas Emendas Constitucionais nº 1/1992 a 95/2016 e pelo Decreto Legislativo nº 186/2008. Brasília: Senado Federal, Coordenação de Edições Técnicas, 2016a.

_____. **Constitution of the Federative Republic of Brazil**: constitutional text enacted on October 5, 1988, with the alterations established by Revision Constitutional Amendments No. 1, 1994 through 6, 1994, by Constitutional Amendments No.1, 1992 through 92, 2016, and by Legislative Decree No. 186, 2008. 5th. ed. Brasília: Edições Câmara, 2016b. Disponível em: http://bd.camara.gov.br/bd/bitstream/handle/bdcamara/36019/constitution_ing-5ed.pdf?sequence=1. Acesso em: 11 nov. 2018.

_____. **Lei nº 11.637 de 28 de dezembro de 2007**. Altera e revoga dispositivos da Lei no 6.404, de 15 de dezembro de 1976, e da Lei no 6.385, de 7 de dezembro de 1976, e estende às sociedades de grande porte disposições relativas à elaboração e divulgação de demonstrações financeiras. Brasília, DF: Presidência da República, 2007. Disponível em: http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2007/lei/l11638.htm. Acesso em: 30 dez. 2018.

BRASIL. Câmara dos Deputados. **Resolução da Câmara dos Deputados nº 28 de 22 de junho de 1955**. Ementa: Reconhece, como serviço de cooperação interparlamentar, o Grupo Brasileiro filiado à União Interparlamentar, sediada em Genebra. Rio de Janeiro: Diário do Congresso Nacional, 1955a. Disponível em: http://legis.senado.leg.br/norma/559759/publicacao/15731732. Acesso em: 1 jun. 2019.

BRASIL. Senado Federal. **Resolução do Senado Federal nº 9 de 06 de junho de 1955**. Ementa: Reconhece, como serviço de cooperação interparlamentar, o Grupo Brasileiro filiado à União Interparlamentar, sediada em Genebra. Rio de Janeiro: Diário do Congresso Nacional, 1955b. Disponível em:

http://legis.senado.leg.br/norma/589980/publicacao/15835671. Acesso em: 1 jun. 2019.

BRASIL. **Tradução do Glossário de Termos Legislativos do Congresso Nacional**. Tradução em inglês, francês e espanhol. Brasília: Senado Federal, 2019.

BRITZ, Denny *et al*. **Massive exploration of neural machine translation architectures**. 2017. Disponível em: https://arxiv.org/pdf/1703.03906.pdf. Acesso em: 30 set. 2018.

CEPERO, Andrés Herrera *et al*. **Sentence Fusion Supervised Project**. [S.l.]: Université de Lorraine , 2018. Disponível em: http://institut-sciences-digitales.fr/wp-content/uploads/2018/06/IDMC-2017_2018-ProjetTutore-RAPPORT-_herrera-jeannot-pouvreau-zouhri.pdf. Acesso em: 28 set. 2018

CHEN, Mia Xu; FIRAT, Orhan; BAPNA, Ankur. **The Best of BothWorlds:** Combining Recent Advances in Neural Machine Translation. 2018. Disponível em: https://arxiv.org/pdf/1804.09849v2.pdf. Acesso em: 10 jan. 2019.

CHO, Kyunghyun; MERRIËNBOER, Bart van; BAHDANAU, Dzmitry. **On the properties of neural machine translation:** Encoder-decoder approaches. 2014. Disponível em: https://arxiv.org/pdf/1409.1259.pdf. Acesso em: 7 set. 2018.

COLLOBERT, Ronan *et al*. Natural language processing (almost) from scratch. **Journal of Machine Learning Research**, p. 2493-2537, 12 Aug. 2011. Disponível em: http://www.jmlr.org/papers/v12/collobert11a.html. Acesso em: 7 set. 2018.

COULTHARD, Malcolm; CALDAS-COULTHARD, Carmen Rosa. **Tradução:** teoria e prática. Florianópolis: Editora da UFSC, 1991. 224 p. Disponível em: http://murieltranslations.com/articles/machine_translation/a_traduc_autom-coulthard.pdf. Acesso em: 15 dez. 2018.

DABBISH, Laura *et al*. **Social coding in GitHub**: transparency and collaboration in an open software repository. *In*: CONFERENCE ON COMPUTER SUPPORTED COOPERATIVE WORK, 2012, Seattle. **Proceedings** […]. Seattle: ACM, 2012, p. 1277-1286. Disponível em: https://dl.acm.org/citation.cfm?id=2145396. Acesso em: 29 set. 2018.

DANIELSON, Donald. **Vectors and tensors in engineering and physics**. 2ª ed. Boca Raton: CRC Press, 2018.

DAUPHIN, Yann N. *et al*. **Language Modeling with Gated Linear Units**. 2017. Disponível em: https://arxiv.org/pdf/1612.08083v3.pdf. Acesso em: 9 set. 2018.

THE ECONOMIST GROUP LIMITED. The machine of a new soul; Neuromorphic computing. **The Economist**, Science and Tecnology, London, v. 408, n. 8847, p. 67-n/a, 3 Ago. 2013.

ERJAVEC, Tomaz. MULTEXT-East Version 3: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. *In*: INTERNATIONAL CONFERENCE ON LANGUAGE RESOURCES AND EVALUATION, 4., 2004, Lisbon. **Proceedings [**…]. Lisbon: European Language Resources Association, 2004, v.4, p. 1535-1538. Disponível em: http://lrec-conf.org/proceedings/lrec2004/pdf/109.pdf . Acesso em: 6 nov. 2018.

ESTOPACE, Eden. EUR 82 Per Page: Inside the EU Translation Centre's 2018 Budget. **Slator: Language Industry Intelligence**, Zurich, Sept. 2017.

EUROMATRIX. **Evaluation Matrix**: Translation quality of best system for test set. 2018. Disponível em: http://matrix.statmt.org/matrix. Acesso em: 11 jan. 2019.

FEIGENBAUM, Edward A. Expert systems in the 1980s. **State of the art report on machine intelligence.** Maidenhead: Pergamon-Infotech, 1981. Disponível em: https://stacks.stanford.edu/file/druid:vf069sz9374/vf069sz9374.pdf. Acesso em: 19 dez. 2018.

FERREIRA, M. Coutinho. Campos léxico-semânticos e o ensino de vocabulário de segunda língua. **Revista Prolíngua**, v. 2, n. 2, 2009.

FREITAS, Eduardo A Mello. **Criação de uma de via de mão dupla para o conhecimento do processo legislativo da Câmara dos Deputados**: relatório de intervenção. Brasília: Câmara dos Deputados, 2017, 36 f.

FUKUSHIMA, Kunihiko. A Feature Extractor for Curvilinear Patterns: A Design Suggested by the Mammalian Visual System. **Kybernetic 7**, n.4, p. 153-160, 1970. Disponível em: https://link.springer.com/article/10.1007/BF00571695. Acesso em: 30 dez. 2018.

GEHRING, Jonas *et al*. A Convolutional Encoder Model for Neural Machine Translation. **Facebook AI Research**. 2016. Disponível em: https://arxiv.org/pdf/1611.02344.pdf. Acesso em: 7 set. 2018.

GEHRING, Jonas *et al*. Convolutional Sequence to Sequence Learning. **Facebook AI Research**. 2017. Disponível em: https://arxiv.org/pdf/ 1705.03122v3.pdf. Acesso em: 20 dez. 2018.

GUDWIN, Ricardo. **Introdução aos Sistemas Inteligentes**. Campinas, SP: UNICAMP, 2003. 162 slides. Disponível em: https://slideplayer.com.br/slide/5631904. Acesso em: 17 dez. 2018.

HASSAN, Hany *et al*. **Achieving Human Parity on Automatic Chinese to English News Translation**. [S.l.]: Microsoft AI & Research, 2018. Disponível em: https://arxiv.org/pdf/1803.05567v2.pdf. Acesso em: 7 jan. 2019.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long short-term memory. **Neural computation**, v. 9, n. 8, p. 1735-1780, 1997. Disponível em: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.676.4320&rep=rep1&type=pdf . Acesso em: 19 nov. 2018.

HUTCHINS, W. John. First steps in mechanical translation. *In*: MACHINE TRANSLATION SUMMIT, 4., 1997, San Diego. **Proceedings [**…]. Washington: AMTA, 1997, p. 14-23. Disponível em: http://hutchinsweb.me.uk/MTS-1997.pdf. Acesso em 4 nov. 2018.

HUTCHINS, W. John. Machine translation: A concise history. Computer aided translation: Theory and practice, 13, p. 29-70, 2017. Disponível em: http://www.hutchinsweb.me.uk/CUHK-2006.pdf. Acesso em: 7 dez. 2018.

INSTITUTO BRASILEIRO DE PLANEJAMENTO E TRIBUTAÇÃO. Quantidade de Normas Editadas no Brasil: 28 anos da Constituição Federal de 1988. Curitiba: IBPT, 2016. Disponível em: https://www.conjur.com.br/dl/estudo-ibpt-edicao-criacao-leis.pdf. Acesso em: 7 set. 2018.

INTER-PARLIAMENTARY UNION. **Annual report on the activities of the Inter-Parliamentary Union**. Geneva: IPU, 2017. Disponível em: https://www.ipu.org/resources/publications/about-ipu/2018-03/annual-report-activities-inter-parliamentary-union-2017. Acesso em 4 nov. 2018.

_____. **Inter-Parliamentary Open Data Cloud**. A project within the Centre for Innovation in Parliament. Geneva: IPU, 2019.

_____. **Design and development of an open data platform for IPU and data integration within the IPU.org website**: Request for Porposals. Geneva: IPU, 2016. Disponível em: http://archive.ipu.org/finance-e/rfp_Opendata.pdf . Acesso em: 25 set. 2018.

JOHNSON, Melvin *et al*. **Google's Multilingual Neural Machine Translation System**: Enabling Zero-Shot Translation. 2016. Disponível em: https://arxiv.org/pdf/1611.04558. Acesso em: 28 set. 2018.

KALCHBRENNER, Nal; GREFENSTETTE, Edward; BLUNSOM, Phil. **A Convolutional Neural Network for modeling sentences**. 2014. Disponível em: https://arxiv.org/pdf/1404.2188. Acesso em: 30 dez. 2018.

KIM, Yoon. **Convolutional neural networks for sentence classification**. 2014. Disponível em: https://arxiv.org/pdf/1408.5882. Acesso em: 5 nov. 2018.

KLEIN, Guillaume *et al*. **Opennmt**: Open-source toolkit for neural machine translation. 2017. Disponível em: http://aclweb.org/anthology/P17-4012. Acesso em: 20 set. 2018.

KOEHN, Philipp. **Statistical Machine Translation System**: User Manual and Code Guide. Edinburgh: University of Edinburgh, 2018. Disponível em: http://www.statmt.org/moses/manual/manual.pdf. Acesso em: 28 set. 2018.

KUMAR, Poorna; VENUGOPAL, Viswajith. **Explorations in Identifying and Summarizing Subjective Content in Text**. Stanford: Stanford University, 2016. Disponível em: https://cs224d.stanford.edu/reports/poorna.pdf . Acesso em: 5 abr. 2017.

LI, Fei-Fei; JOHNSON, Justin; YEUNG, Serena. **Computer Vision - Lecture 06**: Training Neural Networks. Palo Alto: Stanford University, 2018. Disponível em: http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture04.pdf. Acesso em: 19 nov. 2018.

LING, Wang; TRANCOSO, Isabel; DYER Chris; BLACK, Alan. **Character-based Neural Machine Translation**. 2015. Disponível em: https://arxiv.org/abs/1511.04586. Acesso em: 20 maio 2019.

LONDON, Michael; HÄUSSER, Michael. Dendric Computation. **Annual Review of Neuroscience**, London, v. 28, p. 503-532, 2005. Disponível em: https://doi.org/10.1146/annurev.neuro.28.061604.135703. Acesso em: 20 dez. 2018.

LUONG, Minh-Tang; PHAM, Hieu; MANNING, Christopher D. **Effective approaches to attention-based neural machine translation**. 2015. Disponível em: https://arxiv.org/abs/1508.04025. Acesso em: 15 dez. 2018.

MARKIEWICZ, Tom; ZHENG, Josh. **Getting Started with Artificial Intelligence**: A Practical Guide to Building Enterprise Applications. Sebastopol: O'Reilly Media, 2017.

MARTINS, Andre; ASTUDILLO, Ramon. From softmax to sparsemax: A sparse model of attention and multi-label classification. *In*: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 33., 2016, New York. **Proceedings** […]. New York: PLMR, 2016, p. 1614-1623, 2016. Disponível em: http://www.jmlr.org/proceedings/papers/v48/martins16.pdf. Acesso em: 4 de jun. 2019.

MARTINS, Ronaldo Teixeira. Tradução automática. **Todas as Letras**, São Paulo, v. 10, n. 2, p.148-169, 2009. Disponível em: http://editorarevistas.mackenzie.br/index.php/tl/article/download/455/271. Acesso em: 15 dez. 2018.

MARUF, Sameen; HAFFARI, Gholamreza. **Document Context Neural Machine Translation with Memory Networks**. 2017. Disponível em: https://arxiv.org/pdf/1711.03688. Acesso em: 5 nov. 2018.

MCCARTHY, John *et al*. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955. **AI Magazine**, v. 27, n. 4, p. 12-14, 2006, Palo Alto. Disponível em: https://www.aaai.org/ojs/index.php/aimagazine/article/view/1904/1802 . Acesso em: 5 nov.2018.

MIKOLOV, Tomas *et al*. Distributed representations of words and phrases and their compositionality. *In*: NEURAL INFORMATION PROCESSING SYSTEMS, 26., 2013, Lake Tahoe. [Advances in Neural Information Processing Systems]. **Proceedings** […]. New York: Curran Associates, 2013a, p. 3111-3119. Disponível em: https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf. Acesso em: 30 set. 2018.

MIKOLOV, Tomas *et al*. **Efficient Estimation of Word Representations in Vector Space**. 2013b. Disponível em: https://arxiv.org/pdf/1301.3781v3. Acesso em: 5 nov. 2018.

MIKOLOV, Tomas; YIH, Wen-tau; ZWEIG, Geoffrey. Linguistic regularities in continuous space word representations. *In*: CONFERENCE OF THE NORTH AMERICAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 2013, Atlanta. **Proceedings** […]. Atlanta: Association for Computational Linguistics, 2013, p. 746-751. Disponível em: https://www.aclweb.org/anthology/N13-1090. Acesso em: 30 set. 2018.

NAVIGLI, Roberto; PONZETTO, Simone Paolo. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. **Artificial Intelligence**, Rome, v. 193, p. 217-250, 2012. Disponível em: http://wwwusers.di.uniroma1.it/~navigli/pubs/AIJ_2012_Navigli_Ponzetto.pdf. Acesso em: 30 set. 2018.

NONAKA, I; TAKEUCHI, H. **Criação de conhecimento na empresa**. Tradução de Ana Beatriz Rodrigues e Priscilla Martins Celeste. Rio de Janeiro: Campus, 1997.

OLAH, Christopher. **Understanding LSTM Networks**. 2015. Disponível em: htpp://colah.github.io/post/2015-08-Understanding-LSTMs/. Acesso em: 1 jan. 2019.

OPENNMT.NET. **OpenNMT Training Models**. [*S. l.: s.n.*]. Disponível em: http://opennmt.net/OpenNMT/training/models/. Acesso em: 12 nov. 2018.

OPENNMT-PY. **Projeto OpenNMT-Py no GitHub**. [*S. l.: s.n.*], 2017. Disponível em: https://github.com/OpenNMT/OpenNMT-py/. Acesso em: 4 jul. 2019.

PAPINENI, Kishore *et al*. BLEU: a Method for Automatic Evaluation of Machine Translation. *In*: ANNUAL MEETING ON ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 2002, Philadelphia. **Proceedings** […].Philadelphia: Association for Computational Linguistics, 2002, p. 311-318. Disponível em: https://dl.acm.org/citation.cfm?id=1073135. Acesso em: 30 set. 2018.

POMBO, Olga. **Leibniz e o problema de uma língua universal**. Lisboa: Junta Portuguesa de Investigação Científica e Tecnológica, 1997.

PORTUGAL. **Base de Termos da Assembleia da República de Portugal**. Lisboa: Assembleia da República de Portugal, 2019.

PROJECT MANAGEMENT INSTITUTE, Inc. **Um guia do conhecimento do Gerenciamento de Projetos (Guia PMBOK®)**. 4. ed. Newtown Square: PMI, 2009.

REITER, Ehud. A Structured Review of the Validity of BLEU. **Computational Linguistics**, v. 44, p. 393-401, 2018. Disponível em: https://www.mitpressjournals.org/doi/abs/10.1162/COLI_a_00322. Acesso em: 07 nov. 2018.

ROQUE, Antonio Carlos. **Psicologia Conexionista**. Ribeirão Preto: Universidade de São Paulo, 2016. Disponível em: http://sisne.org/Disciplinas/PosGrad/PsicoConex/aula3.pdf. Acesso em: 10 dez. 2018.

ROSENBLATT, Frank. **Principles of Neurodynamics**: Perceptrons and the Theory of Brain Mechanisms. Washington D.C.: Spartan Books, 1986.

SCHERER, Amanda E.; KADER, Carla C. Os aspectos linguísticos da tradução à luz dos pressupostos teóricos de Roman Jakobson versus a vertente da tradução da linguística de *corpus*. **Entretextos**, Londrina, v.12, n.1, p.132-148, 2012.

SCHMIDHUBER, Jürgen. Deep learning in neural networks: An overview. *In*: **Neural networks**, v. 61, p. 85-117, 2015. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0893608014002135. Acesso em: 11 nov. 2018.

SENELLART, Jean *et al*. OpenNMT System Description for WNMT 2018: 800 words/sec on a single-core CPU. *In*: WORKSHOP ON NEURAL MACHINE TRANSLATION AND GENERATION, 2., 2018, Melbourne. **Proceedings** […]. Melbourne: Association for Computational Linguistics, 2018, p. 122-128. Disponível em: https://www.aclweb.org/anthology/W18-27.pdf . Acesso em 11 nov. 2018.

SENNRICH, Rico; HADDOW, Barry. **Linguistic input features improve neural machine translation**. 2016. Disponível em: https://arxiv.org/pdf/1606.02892.pdf. Acesso em: 11 nov. 2018.

SENNRICH, Rico; HADDOW, Barry; BIRCH, Alexandra. Neural machine translation of rare words with subword units. 2015. Disponível em: https://arxiv.org/pdf/1508.07909.pdf. Acesso em 11 nov. 2018.

SIMONS, G.F.; FENNING, C.D. **Ethnologue**: Languages of the World. Dallas: SIL International, 2018. Disponível em: http://www.ethnologue.com Acesso em: 25 set. 2018.

SMITH, Lindsay I. **A tutorial on principal components analysis**. Relatório Técnico (Ciência da Computação) – Department of Computer Science, University of Otago, Dunedin, 2002.

SQUARTINI, Nicola. **Tensores**. Tutorial sobre o uso de tensores com biblioteca de software hackage, versão 0.3.0.1, 2013. Disponível em: http://noaxiom.org/tensor. (acesso em 10 out. 2018).

SRIVASTAVA, N. *et al*. Dropout: a simple way to prevent neural networks from overfitting. **The journal of machine learning research**, n.15, p.1929-1958, 2014. Disponível em: http://jmlr.org/papers/v15/srivastava14a.html. Acesso em: 4 jul. 2019.

STANFORD WIKI. **Convolution**. [S. l.]: Stanford University, 2018. Disponível em: http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution. Acesso em: 11 nov. 2018.

STRUKOV, D.B. Smart connections. **Nature**, London, v. 476, n. 7361, p. 403-405, Aug 25 2011.

SUÍÇA. [Constituição 1999] **Constitution fédérale de la Confédération suisse**, du 18 avril 1999 (Etat le 1er janvier 2018). Berna, [2018]. Disponível em: https://www.admin.ch/ch/f/rs/1/101.fr.pdf. Acesso em: 11 nov. 2018.

SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. *In*: NEURAL INFORMATION PROCESSING SYSTEMS, 27., 2014, Montreal. [Advances in Neural Information Processing Systems]. **Proceedings** […]. Montreal: NIPS, 2014, p. 3104-3112. Disponível em: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf. Acesso em 06 abr. 2019.

TIEDEMANN, Jörg. Character-Based Pivot Translation for Under-Resourced Languages and Domains. *In*: CONFERENCE OF THE EUROPEAN CHAPTER OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS, 13., 2012, Avignon. **Proceedings** […].Avignon: Association for Computational Linguistics, 2012 p.141-151. Disponível em: https://www.aclweb.org/anthology/E12-1.pdf. Acesso em: 30 set. 2018.

TU, Zhaopeng *et al*. **Multi-Head Attention with Disagreement Regularization**. 2018. Disponível em: https://arxiv.org/pdf/1810.10183.pdf . Acesso em: 11 jan. 2019.

UNIÃO EUROPEIA. **Technological Innovation and Democracy**. Brussels: The General Secretariat of the European Parliament, 2017.

UNIÃO EUROPEIA. **EUR-Lex**. [*S. l.*]: União Europeia, 2018. Disponível em: https://eur-lex.europa.eu/homepage.html. Acesso em: 18 jun. 2018a.

_____. **Single Market**: 25 years of the EU single Market. [*S.l.*]: European Union, 2018. Disponível em: https://europa.eu/european-union/topics/single-market_en. Acesso em: 25 set. 2018b.

_____. **Interpreting and translating for Europe**. [*S.l.*]: Directorate General for Interpretation, 2006. Disponível em: https://ec.europa.eu/info/sites/info/files/en_print_2016.pdf. Acesso em: 14 nov. 2018.

VANIN, Aline A. Língua, cognição e cultura: uma relação indissociável. **Letrônica**, Porto Alegre, v.2, n.1, p. 42-59, 2009. Disponível em: http://revistaseletronicas.pucrs.br/ojs/index.php/letronica/article/4992/4040. Acesso em: 10 jan. 2019.

VASWANI, Ashish *et al*. Attention is all you need. *In*: NEURAL INFORMATION PROCESSING SYSTEMS, 2017, Long Beach. [Advances in Neural Information Processing Systems]. **Proceedings** […]. Long Beach: NIPS, 2017, p. 5998-6008. Disponível em: https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf. Acesso em: 11 nov. 2018.

VICHESSI, Beatriz. Qual a diferença entre língua, idioma e dialeto? **Nova Escola** [Online], São Paulo, 7 mar. 2018. Disponível em: https://novaescola.org.br/conteudo/230/qual-diferenca-lingua-idioma-dialeto. Acesso em: 3 dez. 2018.

VON BARTHELD, C. S.; BAHNEY, J.; HERCULANO-HOUZEL, S. The search for true numbers of neurons and glial cells in the human brain: A review of 150 years of cell counting. **Journal of Comparative Neurolology**, v. 524, n. 18, p. 3865–3895, 2016. Disponível em: https://doi.org/10.1002/cne.24040. Acesso em: 30 set. 2018.

VYGOTSKY, Lev Semenovich. **Pensamento e Linguagem**. Edição eletrônica: Ed. Ridendo Castigat Mores, 2001. *E-book*. Disponível em: http://www.jahr.org. Acesso em: 5 nov. 2018.

WANG, Zhining; WANG, Nianxin. Knowledge sharing, innovation and firm performance. **Expert systems with applications**, v. 39, n. 10, p. 8899-8908, 2012. Disponível em: https://doi.org/10.1016/j.eswa.2012.02.017. Acesso em: 10 jan. 2019.

WORD2VEC. **Tool for computing continuous distributed representations of words**. [Open Source Project]. [*S. l.*: *s. n.*], 2013. Disponível em: https://code.google.com/p/word2vec/. Acesso em: 30 set. 2018.

WU, Yonghui *et al*. **Google's neural machine translation system**: Bridging the gap between human and machine translation. 2016. Disponível em: https://arxiv.org/abs/1609.08144 . Acesso em: 11 nov. 2018.

ZAREMBA, Wojciech; SUTSKEVER, Ilya; VINYALS, Oriol. **Recurrent neural network regularization**. 2014. Disponível em: https://arxiv.org/pdf/1409.2329.pdf. Acesso em 29 set. 2018.

APÊNDICE A – ALTERAÇÕES NO CÓDIGO DO OPENNMT-PY E PYTORCH

Para entendimento deste conteúdo, são necessários conhecimentos de linguagem de programação, especialmente Python. Nem todo o desenvolvimento está explícito no presente documento, uma vez que o material ficaria muito extenso. O destaque foi dado às alterações nos códigos dos projetos abertos OpenNMT-py e PyTorch, as quais foram fruto de:

- Tratamento de erros encontrados na execução do projeto.
- Aproveitamento de parâmetros ainda constantes na documentação do projeto, mas que de fato não eram usados.
- Criação de novos parâmetros e filtros.
- Instalação do serviço web.
- Adequação da versão OpenNMT-py para fazer uso de recursos usados na versão original, escrita em Lua e migração do uso dos tokenizadores Perl para Python.

1. Tratamento de erros

O erro que gerou mais dúvida na resolução foi o da classe denominada batch (arquivo batch.py do pacote torchtext). Ele completa o PyTorch no processamento de linguagem natural. O início do arquivo possui o seguinte conteúdo[2]:

```
import torch

class Batch(object):
    """Defines a batch of examples along with its Fields.
    Attributes:
        batch_size: Number of examples in the batch.
        dataset: A reference to the dataset object the examples come from
            (which itself contains the dataset's Field objects).
        train: Deprecated: this attribute is left for backwards compatibility,
            however it is UNUSED as of the merger with pytorch 0.4.
        input_fields: The names of the fields that are used as input for the model
        target_fields: The names of the fields that are used as targets during
                       model training

    Also stores the Variable for each column in the batch as an attribute.
    """
    def __init__(self, data=None, dataset=None, device=None):
        """Create a Batch from a list of examples."""
        if data is not None:
            self.batch_size = len(data)
            self.dataset = dataset
            self.fields = dataset.fields.keys()  # copy field names
            self.input_fields = [k for k, v in dataset.fields.items() if
                                 v is not None and not v.is_target]
```

[2] Algumas linhas em branco foram removidas para melhorar a visualização.

```
            self.target_fields = [k for k, v in dataset.fields.items() if
                                  v is not None and v.is_target]
```

A classe Batch recebe a estrutura de um conjunto de dados e os dados correspondentes. Os dados podem ser os de entrada ou os esperados na saída. O problema foi identificado no momento da tradução pelo serviço web. O ambiente incluía Python 3.7, PyTorch 0.4.1 (com a biblioteca "torchtext" correspondente) e a versão 0.7 da OpenNMT-py como atualização da base do tradutor. Nesse caso, apenas a entrada de dados em Português era enviada pela página web a fim de receber a tradução em Inglês. Só haveria a saída na forma de predição. Então, se a predição viesse vazia, não havendo tratamento de erro específico ou identificação do tipo de dado, o Python teria de considerar a condicional `v.is_target` como sendo falsa. Porém, não é seguro garantir-se em um comportamento específico do interpretador ou compilador, nem esperar que algo feito para lidar com diferentes tipos de dados não precise de verificação, pois pode bastar uma mudança de versão para se receber uma mensagem de erro. A mensagem foi: "object has no attribute 'is_target' ".

Como a estrutura do loop e a condicional para o loop estão na mesma linha. Preferiu-se evitar a quebrar da estrutura para simplificar a ação de mesclagem de versões de código. A solução, então, enquanto se corrigia o problema, foi a criação de uma função para substituir `v.is_target`. Não havendo valor, o campo seria entendido como falso, caso contrário, o valor seria retornado. Uma simples função que foi denominada `check_is_target` passou a receber o campo e verificar o atributo `is_target`.

```
   self.input_fields = [k for k, v in dataset.fields.items() if
                        v is not None and not check_is_target(v)]
    self.target_fields = [k for k, v in dataset.fields.items() if
                        v is not None and check_is_target(v)]
```

Em caso de erro, o valor "False" é retornado.

```
def check_is_target(argumento):
    try:
        if argumento.is_target(): return argumento.is_target()
    except:
       return False
```

Alternativa foi se assegurar que todos os tipos de dados recebessem algum valor padrão. Apesar de o erro ter sido corrigido, foi mantida uma cópia da versão com a correção, caso volte o problema.

A evolução das versões, algumas vezes, traz novos problemas. Um deles foi que o projeto OpenNMT-py passou a não se comportar bem com a utilização de propriedades externas que geralmente eram concatenadas. Este fato ficou mais notório com os passos da

migração do PyTorch: de 0.4 para 0.41 e em seguida 1.0.0. Neste caso, existem comentários internos no código que informam que não irá funcionar com a informação o uso de letras maiúsculas ou minúsculas, o que afastou o projeto da versão original. Foi revisado o código em "onmt /inputters/dataset_base.py", um pedaço da extração está assim descrita no código:

```
specials = [PAD_WORD, UNK_WORD, BOS_WORD, EOS_WORD]
words = []
features = []
n_feats = None
for token in tokens:
    #split_token = token.split(u"|")
    split_token = token.split(u"\uffe8")
    assert all([special != split_token[0] for special in specials]), \
             "Dataset cannot contain Special Tokens"
```

A alteração discutida não foi o bastante, pois para cada sentença, ao final, o tamanho das matrizes de cada batch do cálculo da predição, ao final da sentença, vinha com tamanho fixo e provocava um erro ocorre ao final da sentença. Foi iniciada a revisão dos cálculos, o que apontou para o erro nas dimensões do último elemento do fim de sentença. Dado o grande trabalho envolvido, a opção recaiu sobre uma solução de contorno, que foi a de incluir um *token* informando quando a palavra viesse a ter uma letra ou mais em maiúsculo. Porém, qualquer que seja a solução, algumas vezes o tradutor humano muda a capitalização não por conta das características próprias das línguas, como é o caso dos nomes dos meses escritos em maiúsculo. Mas o problema pode ser por conta de efeitos estéticos, o que confunde o tradutor.

Certos comportamentos podem ser considerados erro não esperado, tal como uma linha apenas com um caractere de nova linha, isto é, uma linha vazia. Contudo, o problema foi resolvido no tratamento que antecede a tradução de modo a evitar o erro. No caso do treinamento, a tokenização passou a poder receber dois textos alinhados, ou paralelos, um com a língua de entrada e outro com a saída. O tokenizador criado recebe ambos e trata os textos ao mesmo tempo e, dentre outras coisas, verifica se algum deles possui linha vazia. No caso de uma das linhas estar vazia, ambas deverão deixar de ser usadas, visto se tratar de um erro e utilizá-las levará a erro no condução do treinamento. Linhas filtradas são gravadas em arquivo separado para avaliação.

2. Hiperparâmetros ignorados

Alguns hiperparâmetros estavam sendo completamente ignorados. Isso pode ser considerado um erro, ainda que não haja mensagem alerta para ele. O parâmetro `lower`, que

voltou a ser usado pela versão mestre do OpenNMT-py, indica que o pré-processamento irá guardar no vocabulário apenas palavras em minúsculo. Isso não significa que será impossível traduzir palavras com letras em maiúsculo, visto que é possível fazer uso de *tokens* para identificar o tipo de letra usada, ou ainda incluir propriedade com tal informação, a qual seria concatenada com o vetor correspondente à palavra ou letra.

Outro parâmetro é o `save_data`, que indica um caminho para guardar o vocabulário. Nesse caso, o programa foi alterado para guardar o vacabulário em arquivos de texto não formatados, um para cada língua, a fim de permitir a conferência dos termos tratados, seja pelos filtros, pelas proteções ou pela tokenização. Os arquivos alterados foram "preprocess.py" e "onmt/inputters/inputter.py" passaram a conter filtros com expressões regulares.

Há um conjunto de hiperparâmetros que não se comporta como esperado quando informado o valor zero. De acordo com a documentação, a definição de zero para o limite no número de palavras, deveria corresponder à ausência de limite. Contudo, ao invés de contar todas as palavras, apenas alguns tokens são contados. Após a correção, o número de total de palavras passou a ser considerado sem limite no vocabulário.

3. Criação de novos hiperparâmetros e filtros

Também foi criado o parâmetro denominado de `only_words`, que pode ter mais de um valor. Se o valor for 1, então, apenas palavras com letras do alfabeto latino e hífen serão consideradas como vocabulário. Se for utilizado o 2, então outros símbolos são incluídos.

Parâmetros novos, no projeto OpenNMT-py, devem ser incluídos no arquivo de configurações.. Segue um exemplo do código do arquivo "opts.py", onde na cor em vermelho está um dos parâmetros criados:

```
# Truncation options, for text corpus
group = parser.add_argument_group('Pruning')
group.add('--src_seq_length', '-src_seq_length', type=int, default=50,
        help="Maximum source sequence length")
group.add('--src_seq_length_trunc', '-src_seq_length_trunc',
        type=int, default=None,
        help="Truncate source sequence length.")
group.add('--tgt_seq_length', '-tgt_seq_length', type=int, default=50,
        help="Maximum target sequence length to keep.")
group.add('--tgt_seq_length_trunc', '-tgt_seq_length_trunc',
        type=int, default=None,
        help="Truncate target sequence length.")
```

```
group.add('--lower', '-lower', action='store_true', help='Lowercase data')
group.add('--only_words', '-only_words',
        type=int, default=0, help="1 = only words; 2 = words and puctuation marks.")
group.add('--filter_valid', '-filter_valid', action='store_true',
        help='Filter validation data by src and/or tgt length')
```

Juntamente com o novo parâmetro deve-se indicar o tipo de dado, o valor padrão no caso de não ser dada atribuição na chamada do programa e uma descrição do parâmetro como ajuda. No caso de erro na atribuição do valor do parâmetro, a descrição será apresentada e servirá de ajuda. As opções obedecem a uma hierarquia, de modo que a opção deve fazer parte de um grupo de parâmetros. O parâmetro `only_words` foi criado como forma de restringir o vocabulário.

4. Adequação e migração de programas utilitários para Python

Verificou-se que uma das bibliotecas que podem ser utilizadas para a tokenização no serviço web é praticamente idêntica ao utilitário de tokenização da versão OpenNMT-lua. Por este motivo, optou-se por construir um novo tokenizador em Python com uso desta biblioteca, a `pyonmttok`. O Torch foi descontinuado e, por conta disso, o projeto OpenNMT-lua também o foi. Na etapa anterior ao pré-processamento, chegou-se a usar um programa em Perl. Ao final, todos esses programas foram substituídos pelos seguintes programas:

a) protect_token3.py – programa que faz a proteção de *tokens* conforme definição presente em filtros de expressões regulares. Identificação de Leis, números de telefone, e-mail, *hashtag*, endereço url, códigos do Parlamento Europeu, repetição de caracteres tais como pontos e hifens repetidos, dentre outros filtros. Em última análise, os formatos das leis e a substituição direta de tradução estão sendo avaliadas, especialmente porque várias leis foram obtidas sem a respectiva formatação. Os *tokens* protegidos devem passar direto pelo processo de tradução. Ao final eles são reconstruídos a partir das informações armazenadas em arquivo. Ao ser executado, o programa cria uma proteção com identificação que deverá passar livre pelo tokenizador e pelo tradutor praticamente sem interferência.
b) unpack_token2.py – programa que substitui as proteções pelos termos que foram protegidos e guardados previamente no arquivo gerado pelo programa protect_token.py.

c) tokeduamf.py – programa que faz a tokenização com parâmetros idênticos aos usados pelo serviço web do protótipo. Enviado um documento, o programa irá identificar palavras candidatas à proteção. A tokenização é feita com base em palavras ou caracteres e é usado na tradução de documentos em lote.
d) tokeduamf_duo.py – semelhante ao anterior, porém, por receber dois textos de entrada que devem estar alinhados entre si, a presença de discrepância, tal como uma linha vazia em uma língua e conteúdo na outra, faz com que tal linha não seja utilizada. Seu uso visa o processamento para o uso em treinamento, validação ou ainda teste de qualidade.
e) detokeduamf.py – programa que restaura a tokenização dos programas tokeduamf*.

5. Serviço web

O serviço web já vem preparado, mas também apresenta problemas de parâmetros apresentados que não funcionam. O principal deles para apresentação na página web, que é o nome do modelo, não era tratado, isto é, o parâmetro `name` era totalmente ignorado. O código foi mudado para tratar o nome do modelo e identificar o tipo de tradutor. Os valores a serem passados como opções tiveram de ser descobertos a partir dos parâmetros possíveis de tokenização, informação esta que já foi compartilhada nos fóruns em resposta às dúvidas de outros usuários e desenvolvedores do OpenNMT-py.

A configuração do serviço em relação a como deve ser usado o modelo é feita da seguinte forma:

1) Configuram-se os parâmetros do serviço que usará o Flask. Os parâmetros também podem ser passados por linha de comando;
2) Definem-se os modelos no arquivo de configuração do serviço, normalmente "conf.json", que também irá conter as informações para a tokenização e para a tradução;
3) Criação de uma página web para acessar o serviço na forma de uma REST API. Para informações e pequenas ações o serviço receberá requisições do tipo GET (lista de modelos e parâmetros, por exemplo) e do tipo POST (no envio do texto a ser traduzido e e recebimento da tradução). Linhas de comando podem ser usadas para verificar o funcionamento.

Cada modelo tem sua identificação própria, de forma que é possível ativar e desativar um modelo por comando enviado pela página web. É comum determinar que, após

um determinado tempo sem utilização do tradutor, o modelo seja copiado para se manter carregado em memória acessada pela CPU. A seguir, um exemplo adaptado da configuração usada pelo serviço de tradução automática pela web “conf.json” pelo protótipo.

```
{
    "models_root": "./available_models",
    "models": [
        {
            "id": 101,
            "name": "PT-EN (bi Long Short-Term Memory, Recurrent Neural Network)",
            "model": "BiLSTM2-stok_2019_05_27-07_30__step_795000.pt",
            "dynamic_dict": true,
            "timeout": 1000,
            "on_timeout": "to_cpu",
            "model_root": "/opt/models",
            "load": true,
            "opt": {
                "gpu": 0,
                "beam_size": 5,
                "max_length": 650,
                "batch_size": 32,
                "share_vocab": true,
                "replace_unk": true,
                "verbose": true
            },
            "tokenizer": {
                "type": "pyonmttok",
                "mode": "aggressive",
                "params": {
                    "no_substitution": false,
                    "spacer_annotate": false,
                    "spacer_new": false,
                    "joiner_annotate": true,
                    "joiner_new": false,
                    "case_markup": true,
                    "case_feature": false,
                    "preserve_placeholders": true,
                    "preserve_segmented_tokens": true,
                    "segment_case": true,
                    "segment_numbers": true,
                    "segment_alphabet_change": false
                }
            }
        }
    ]
}
```

Convém observar que o valor zero correspondente à GPU não significa que não será usada a GPU. No caso de não ser usada GPU, o valor deve corresponder a -1.

A parte de código a seguir faz parte do arquivo alterado translation_server.py. Ele funciona como interface com o programa de tradução e é acionado pelo serviço online. Com a alteração, o serviço web de tradução passou a quebrar as linhas a partir da marcação de término de linha. O erro estava na perda das marcações de quebra de linha porque o separador foi erroneamente substituído por espaço em branco. Assim sendo as linhas estavam sendo enviadas para a tradução aglutinadas, como uma só sentença. A correção resultou em melhor

tempo de resposta e cuidados nos processos de tokenização e de detokenização. O protótipo passou a ser capaz de processar um grande número de linhas por vez.

Translate_server.py:

```
...
for i, inp in enumerate(inputs):
    src = inp['src']
    if src.strip() == "":
        head_spaces.append(src)
        texts.append("")
        tail_spaces.append("")
    else:
        whitespaces_before, whitespaces_after = "", ""
        match_before = re.search(r'^\s+', src)
        match_after = re.search(r'\s+$', src)
        if match_before is not None:
            whitespaces_before = match_before.group(0)
        if match_after is not None:
            whitespaces_after = match_after.group(0)
        head_spaces.append(whitespaces_before)
        listTexts = []
        for nSource in src.split("\n"):
            tok = self.maybe_tokenize(nSource.strip())
            listTexts.append(tok)
            sslength.append(len(tok.split()))
            tail_spaces.append(whitespaces_after)
        #texts.append(tok)
        texts = listTexts

empty_indices = [i for i, x in enumerate(texts) if x == ""]
texts_to_translate = [x for x in texts if x != ""]
...
listResults = []
for item in results:
    # results = [self.maybe_detokenize(item) for item in results]
    for nResult in item.split("\n"):
        listResults.append(self.maybe_detokenize(nResult))
results = ["\n".join(listResults)]
...
```

Ao final da execução, o programa devolve os dados com as informações de mudança de linha e com um escore correspondente à tradução.

APÊNDICE B – PYTORCH E TENSORES

Este conteúdo se baseia em tutoriais do PyTorch encontrados em https://pytorch.org/tutorials, traduzidos e adaptados para fins de entendimento do conteúdo mais aprofundado do presente relatório, especialmente o Apêndice C, que trata dos cálculos dos indicadores.

Os tensores são matrizes multidimensionais que podem ser criados desde listas do Python lists com pelo uso do método, ou função, torch.Tensor(). Podem ser criados tensores de diferentes tipos.

```
# A função torch.tensor( ) cria um objeto tensor torch.Tensor a partir de um dado.
# O ponto a cada número é para indicar que não se trata de um número inteiro.

V_ListaPython = [1., 2., 3.]
V = torch.tensor(V_ListaPython)
print(V)
```

Saída:

```
tensor([1., 2., 3.])
```

```
# Criação de uma matrix

M_data = [[1., 2., 3.], [4., 5., 6]]
M = torch.tensor(M_data)
print(M)

# Criação de um tensor de 3 dimensões e tamanhos 2x2x2.

T_data = [[[1., 2.], [3., 4.]],
          [[5., 6.], [7., 8.]]]
T = torch.tensor(T_data)
print(T)
```

Saídas:

```
tensor([[1., 2., 3.], [4., 5., 6.]])
tensor([[[1., 2.], [3., 4.], [5., 6.], [7., 8.]]])
```

Para aqueles que já conhecem a programação em Python, pouco é mudado. Isso ocorre porque o PyTorch é um framework dinâmico, que funciona como a linguagem, sendo por isso mais fácil de ser depurado, mais versátil e não depende de um grafo de operações, isto é, a rede é preparada para a execução. O Tensorflow é mais estático, contudo a partir da versão 1.7 a depuração se tornou mais amigável. A vantagem de um framework mais estático é que ele pode aperfeiçoar a execução, tal como em uma compilação.

Já o Keras, ele é mais um recurso de alto nível que usa outros frameworks do que um framework, motivo pelo qual é mais usado pelos iniciantes.

Voltando ao PyTorch, existem coisas que ele faz sozinho quando trabalha com tensores, tais como cálculos de gradiente, diferenciação etc. Ao ser usada uma variável criada com o Numpy, por exemplo, ao invés do PyTorch aguardar receber o valor, ele vai direto à memória e usa os valores diretamente. Assim sendo, o PyTorch agiliza o uso de tensores tanto com CPU como com GPU.

Caso não se queira que o PyTorch acumule dados para fazer cálculos das operações realizadas, deve-se indicar explicitamente. Algumas funções possuem por padrão o cálculo automático como False. Nesse caso, é necessário definir `requires_grad` como True a fim de usar o mecanismo Autograd. A função `torch.randn()` preenche o tensor com valores randômicos dentro de uma distribuição normal. Logo, existirão valores negativos e positivos. No caso de uma distribuição uniforme, os valores irão varia de 0 a 1, o que pode ser obtido com a função `torch.rand()`.

No exemplo a seguir, como x e y não ativaram o mecanismo Autograd, a soma de x e y também não ativa. Contudo, como z ativou o mecanismo, então operações com z também fazem uso do mecanismo.

```
>>> x = torch.randn(5, 5)  # requires_grad=False by default
>>> y = torch.randn(5, 5)  # requires_grad=False by default
>>> z = torch.randn((5, 5), requires_grad=True)
>>> a = x + y
>>> a.requires_grad
False
>>> b = a + z
>>> b.requires_grad
True
```

Caso se queira fazer atualizações de pesos de uma rede usando-se gradiente descendente, por exemplo, não é obrigatório habilitar o mecanismo, basta usar o valor do gradiente para cada peso, o que funciona mesmo com o mecanismo Autograd desligado.

```
# Manually update weights using gradient descent. Wrap in torch.no_grad()
# because weights have requires_grad=True, but we don't need to track this
# in autograd.
# An alternative way is to operate on weight.data and weight.grad.data.
# Recall that tensor.data gives a tensor that shares the storage with
# tensor, but doesn't track history.
# You can also use torch.optim.SGD to achieve this.
with torch.no_grad():
     w1 -= learning_rate * w1.grad
     w2 -= learning_rate * w2.grad

     # Manually zero the gradients after updating weights
     w1.grad.zero_()
     w2.grad.zero_()
```

Para definir o dispositivo como GPU, deve-se primeiro verificar se existe cuda (API para acesso aos núcleos Nvidia) disponível e visível para então definir o dispositivo. Para que um dispositivo esteja visível, recomenda-se a execução da linha de comando: `export CUDA_VISIBLE_DEVICES=0`. O número zero corresponde a um dispositivo. Se forem quatro as GPUs disponíveis, cada uma receberá uma numeração a partir de zero.

A definição de uma semente é muito importante para viabilizar a comparação entre testes e se faz pelo método manual_seed( ) do PyTorch.

```
import torch

torch.cuda.is_available()
torch.cuda.set device(device id) # deve ser o número correspondente à GPU

torch.manual_seed(777)
torch.cuda.manual_seed(777)
```

A criação de uma rede de duas camadas é bem simples. Apesar do aparente tamanho do código, observe que a maior parte trata-se de comentário. A escolha do tipo dos elementos usados (dtype) em um tensor pode alterar o consumo de memória da rede e afetar a precisão numérica.

No código a seguir, utiliza-se a CPU como dispositivo, o número de entradas é de 1000 e o número de saídas é de 10. Observar, que ao se usar a função `clamp`, podem ser atribuídos um valor mínimo e um valor máximo. Se o valor for menor do que o mínimo, o valor retornado será o valor mínimo. Se maior do que o valor máximo, o valor retornado será o valor máximo. Uma vez chamada a função `backward()`, como w1 e w2 requerem o mecanismo, o gradiente é calculado automaticamente.

```
# -*- coding: utf-8 -*-
import torch

dtype = torch.float
device = torch.device("cpu")
# device = torch.device("cuda:0") # Uncomment this to run on GPU

# N is batch size; D_in is input dimension;
# H is hidden dimension; D_out is output dimension.
N, D_in, H, D_out = 64, 1000, 100, 10

# Create random Tensors to hold input and outputs.
# Setting requires_grad=False indicates that we do not need to compute gradients
# with respect to these Tensors during the backward pass.
X = torch.randn(N, D_in, device=device, dtype=dtype)
y = torch.randn(N, D_out, device=device, dtype=dtype)

# Create random Tensors for weights.
# Setting requires_grad=True indicates that we want to compute gradients with
# respect to these Tensors during the backward pass.
w1 = torch.randn(D_in, H, device=device, dtype=dtype, requires_grad=True)
w2 = torch.randn(H, D_out, device=device, dtype=dtype, requires_grad=True)

learning_rate = 1e-6
```

```
for t in range(500):
    # Forward pass: compute predicted y using operations on Tensors; these
    # are exactly the same operations we used to compute the forward pass using
    # Tensors, but we do not need to keep references to intermediate values since
    # we are not implementing the backward pass by hand.
    Y_pred = x.mm(w1).clamp(min=0).mm(w2)

    # Compute and print loss using operations on Tensors.
    # Now loss is a Tensor of shape (1,)
    # loss.item() gets the a scalar value held in the loss.
    Loss = (y_pred - y).pow(2).sum()
    print(t, loss.item())

    # Use autograd to compute the backward pass. This call will compute the
    # gradient of loss with respect to all Tensors with requires_grad=True.
    # After this call w1.grad and w2.grad will be Tensors holding the gradient
    # of the loss with respect to w1 and w2 respectively.
    Loss.backward()

    # Manually update weights using gradient descent. Wrap in torch.no_grad()
    # because weights have requires_grad=True, but we don't need to track this
    # in autograd.
    # An alternative way is to operate on weight.data and weight.grad.data.
    # Recall that tensor.data gives a tensor that shares the storage with
    # tensor, but doesn't track history.
    # You can also use torch.optim.SGD to achieve this.
    With torch.no_grad():
        w1 -= learning_rate * w1.grad
        w2 -= learning_rate * w2.grad

        # Manually zero the gradients after updating weights
        w1.grad.zero_()
        w2.grad.zero_()
```

Caso se queira especificar a função de custo ou objetivo que se deseja usar, isso é possível fazendo uso do pacote "torch.nn". No grupo "torch.nn.funcional" são encontradas funções de ativação e funções de custo, ou perda. Recomendável é a leitura da documentação disponível no sítio do PyTorch em https://pytorch.org/docs/stable/nn.html.

APÊNDICE C – CÁLCULOS DOS PRINCIPAIS INDICADORES

Neste apêndice serão destacados os indicadores mais relevantes para a análise do aprendizado a partir do arquivo “statistics.py” que contém parte dos cálculos dos indicadores estatísticos, a apresentação durante o treinamento e a criação dos dados para o Tensorboard. A classe Statistics, em sua inicialização, recebe o custo (ou perda, ou quanto está do objetivo), o número de palavras processadas e o numero de palavras corretas. Na atualização ele recebe o objeto “stat”, que agrega as informações necessárias calculadas em outros módulos.

```
""" Statistics calculation utility """
from __future__ import division
import time
import math
import sys

from onmt.utils.logging import logger

class Statistics(object):
    """
    Accumulator for loss statistics.
    Currently calculates:
    * accuracy
    * perplexity
    * elapsed time
    """

    def __init__(self, loss=0, n_words=0, n_correct=0):
        self.loss = loss
        self.n_words = n_words
        self.n_correct = n_correct
        self.n_src_words = 0
        self.start_time = time.time()

    @staticmethod
    def all_gather_stats(stat, max_size=4096):
        """
        Gather a `Statistics` object accross multiple process/nodes
        Args:
            stat(:obj:Statistics): the statistics object to gather
                accross all processes/nodes
            max_size(int): max buffer size to use
        Returns:
            `Statistics`, the update stats object
        """
        stats = Statistics.all_gather_stats_list([stat], max_size=max_size)
        return stats[0]

    @staticmethod
    def all_gather_stats_list(stat_list, max_size=4096):
        """
        Gather a `Statistics` list accross all processes/nodes
        Args:
            stat list(list([`Statistics`])): list of statistics objects to
                gather accross all processes/nodes
            max_size(int): max buffer size to use
        Returns:
            our_stats(list([`Statistics`])): list of updated stats
        """
        from torch.distributed import get_rank
        from onmt.utils.distributed import all_gather_list
```

```
    # Get a list of world_size lists with len(stat_list) Statistics objects
    all_stats = all_gather_list(stat_list, max_size=max_size)

    our_rank = get_rank()
    our_stats = all_stats[our_rank]
    for other_rank, stats in enumerate(all_stats):
        if other_rank == our_rank:
            continue
        for i, stat in enumerate(stats):
            our_stats[i].update(stat, update_n_src_words=True)
    return our_stats

def update(self, stat, update_n_src_words=False):
    """
    Update statistics by suming values with another `Statistics` object
    Args:
        stat: another statistic object
        update_n_src_words(bool): whether to update (sum) `n_src_words`
            or not
    """
    self.loss += stat.loss
    self.n_words += stat.n_words
    self.n_correct += stat.n_correct

    if update_n_src_words:
        self.n_src_words += stat.n_src_words

def accuracy(self):
    """ compute accuracy """
    return 100 * (self.n_correct / self.n_words)

def xent(self):
    """ compute cross entropy """
    return self.loss / self.n_words

def ppl(self):
    """ compute perplexity """
    return math.exp(min(self.loss / self.n_words, 100))

def elapsed_time(self):
    """ compute elapsed time """
    return time.time() - self.start_time

def output(self, step, num_steps, learning_rate, start):
    """Write out statistics to stdout.
    Args:
       step (int): current step
       n_batch (int): total batches
       start (int): start time of step.
    """
    t = self.elapsed_time()
    step_fmt = "%2d" % step
    if num_steps > 0:
        step_fmt = "%s/%5d" % (step_fmt, num_steps)
    logger.info(
        ("Step %s; acc: %6.2f; ppl: %5.2f; xent: %4.2f; " +
         "lr: %7.5f; %3.0f/%3.0f tok/s; %6.0f sec")
        % (step_fmt,
           self.accuracy(),
           self.ppl(),
           self.xent(),
           learning_rate,
           self.n_src_words / (t + 1e-5),
           self.n_words / (t + 1e-5),
           time.time() - start))
    sys.stdout.flush()
```

```
def log_tensorboard(self, prefix, writer, learning_rate, step):
    """ display statistics to tensorboard """
    t = self.elapsed_time()
    writer.add_scalar(prefix + "/xent", self.xent(), step)
    writer.add_scalar(prefix + "/ppl", self.ppl(), step)
    writer.add_scalar(prefix + "/accuracy", self.accuracy(), step)
    writer.add_scalar(prefix + "/tgtper", self.n_words / t, step)
    writer.add_scalar(prefix + "/lr", learning_rate, step)
```

Destacando os três principais indicadores:

1) **Acurácia =** 100 * < nº de palavras corretas> / <nº de palavras>
2) **Entropia Cruzada =** < perda> / <nº de palavras>
3) **Perplexidade =** $e^{\min(<perda>/<nº\ palavras>,\ 100)}$

Como a cada atualização é feita a soma da perda. A entropia cruzada calcula o custo médio por palavras durante o treinamento. O PyTorch possui uma função própria para o cálculo do custo com base na entropia cruzada. A função se chama `CrossEntropyLoss`.

O projeto OpenNMT-py, porém, não usa a função `CrossEntropyLoss` pois o cálculo varia de acordo com alguns hiperparâmetros. Juntamente com o arquivo "statistics.py", podem ser encontrados os arquivos "loss.py", "optimizers.py" e "report_manager.py". Uma das funções mais usadas é a nn.NLLLoss (Negative Log-likelihood Loss) do PyTorch combinada com a nn.LogSoftmax, o que resulta na função de custo padrão do PyTorch. O acompanhamento da função de custo é feito a partir dos valores da entropia cruzada.

A entropia cruzada é baseada no log negativo da verossimilhança, que pode ser encontrado a partir do cálculo da perplexidade.

A equação (3) representa o calcula da perplexidade, em que N é o número de palavras (w):

$$ppl = \sqrt[N]{\frac{1}{P(\mathrm{w_1 w_2 w_3 \ldots w_N})}} \qquad (3)$$

A equação (3) pode ser expressa na forma da equação (4).

$$ppl(W) = P(\mathrm{w_1 w_2 w_3 \ldots w_N})^{-1/N} \qquad (4)$$

Como *perplexidade* = $e^{entropia\ cruzada}$, então *entropia* = ln(*perplexidade*).

$$xent(W) = \ln(P(\ w_1 w_2 w_3 \ldots w_N\ )^{-1/N}) \quad (5)$$

O que resulta na equação da entropia cruzada (6), ou seja,

$$xent(W) = -\ln(P(\ w_1 w_2 w_3 \ldots w_N\ )/N \quad (6)$$

Por outro lado, pode-se mostrar a relação entre a entropia cruzada com Softmax, Log_Softmax e NLL (log negativo da verossimilhança) a partir dos seguintes exemplos adaptados do sítio internet do fastai[3] (uma biblioteca usada para ensinar e simplificar o uso de recursos do PyTorch no Python).

```
import torch
import torch.nn as nn
import torch.nn.functional as F
batch_size, n_classes = 5, 3
x = torch.randn(batch_size, n_classes)
x
```

Saída:

tensor([[0.9826, 1.0630, -0.4096],
[-0.6213, 0.2511, 0.5659],
[0.5662, 0.7360, -0.6783],
[-0.4638, -1.4961, -1.0877],
[1.8186, -0.2998, 0.1128]])

```
def softmax(x): return x.exp() / (x.exp().sum(-1)).unsqueeze(-1)
def nll(input, target): return -input[range(target.shape[0]), target].log().mean()

pred = softmax(x)
loss=nll(pred, target)
loss
```

Saída da função de custo a partir da função NLL(softmax(X), Y): tensor(1.4904).

```
def log_softmax(x): return x - x.exp().sum(-1).log().unsqueeze(-1)
def nll(input, target): return -input[range(target.shape[0]), target].mean()

pred = log_softmax(x)
loss = nll(pred, target)
loss

pred_Torch = F.log_softmax(x, dim=-1)
loss_Torch = F.nll_loss(pred_Torch, target)
```

Saídas de ambas as funções NLL(log_softmax(X), Y): tensor(1.4904).

```
F.cross_entropy(x, target)
```

Saída usando a função original do PyTorch: tensor(1.4904).

[3] https://github.com/fastai/fastai_old

A função SparseMaxLoss, do módulo onmt do OpenNMT-py, também pode ser usada como função de custo, onde a Softmax dá lugar à Sparsemax.

Sobre o acompanhamento dos indicadores, no caso de retorno ao treinamento, é importante fazer referência à data de início do treinamento para que haja continuidade no Tensorboard. Trata-se de um exemplo real[4]:

```
CUDA_VISIBLE_DEVICES=0
conda activate pytorch1
cd ~/Projetos/OpenNMT/eduamf/OpenNMT-py
python train.py -data /opt/train/Mix-stok/Brasil_Celex2012-2017 \
 -save_model /opt/models/Brasil_Celex2012-2017_"$(date +"%Y_%m_%d-%I_%M")"_ \
 -seed 777 -gpu_verbose_level 0 -world_size 1 -gpu_ranks 0 -accum_count 1\
 -optim sgd -normalization sents -bridge -residual \
 -copy_attn -global_attention mlp -global_attention_function sparsemax \
 -layers 2 -rnn_size 650 -word_vec_size 600 \
 -rnn_type LSTM -encoder_type brnn -input_feed 1 \
 -valid_steps 1000 -save_checkpoint_steps 2500 -dropout 0.1 -label_smoothing 0.1 \
 -learning_rate 1.2 -learning_rate_decay 0.84 \
 -start_decay_steps 40000 -decay_steps 20000 \
 -log_file logs/treina_"$(date +"%Y_%m_%d-%I_%M")".log -report_every 100 \
 -tensorboard -tensorboard_log_dir runs/onmt/May-14_05-44-04 \
 -batch_size 10 -valid_batch_size 5 \
 -train_steps 800000 \
 -reset_optim keep_states \
 -train_from /opt/models/Brasil_Celex2012-2017_2019_05_27-02_43__step_654000.pt
```

Trata-se do treinamento de uma rede LSTM bidirecional (brnn) onde as palavras foram representadas em 600 dimensões e foram usadas 650 entradas (325 em uma direção e 325 em outra). Alguns parâmetros são para a placa GPU, como é o caso da variável CUDA_VISIBLE_DEVICES e dos parâmetros “world_size”, “gpu_ranks” e “accum_count”. Após um determinado número de passos um arquivo de *checkpoint* é criado conforme o parâmetro “save_checkpoint_steps” e é um backup da RNA contendo informações como a estrutura, os estados e os parâmetros da rede após um determinado número de passos no treinamento. Tanto a tradução como o retorno ao treinamento é feito a partir de um arquivo de *checkpoint*, que representa o modelo resultante do treinamento. No exemplo dado, “keep_states” significa que o estado da rede será o mesmo do *checkpoint*, mas hiperparâmetros de otimização poderão ser alterados. Maiores detalhes podem ser encontrados na documentação do projeto OpenNMT-py.

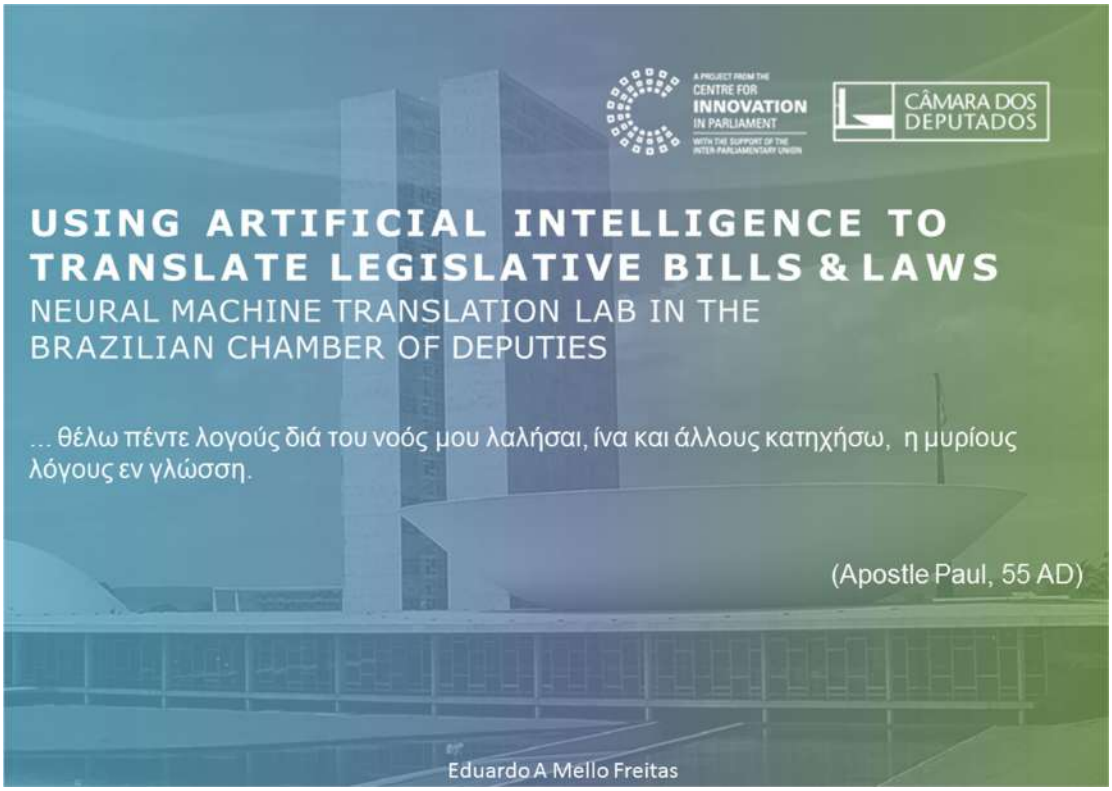
[4] Os nomes de pastas e arquivos foram diminuídos para melhor visualização.

APÊNDICE D – SLIDES DA APRESENTAÇÃO À IPU

Os *slides* a seguir foram apresentados à União Interparlamentar (IPU) na primeira reunião do projeto *Inter-Parliamentary Open Data Cloud*, ocorrida no Brasil, nas dependências da Câmara dos Deputados, no período de 27 a 31 de maio de 2019.

O objetivo do projeto é promover a abertura dos parlamentos em escala global por meio de plataforma digital de modo a prover um ponto único de acesso primário multilíngue às informações legislativas dos países diferentes países. Para tal, devem ser disponibilizados recursos de publicação de dados e se fazer uso de técnicas baseadas em inteligência artificial a fim de automatizar a tradução de leis e a pesquisa aos demais documentos legislativos.

O desenvolvimento e a apresentação dos slides e do protótipo do serviço *online* de tradução automática de leis, Português-Inglês, no ambiente da Intranet da Câmara dos Deputados, fazem parte dos produtos entregues do Curso de Mestrado em Poder Legislativo da Câmara dos Deputados.



Goals

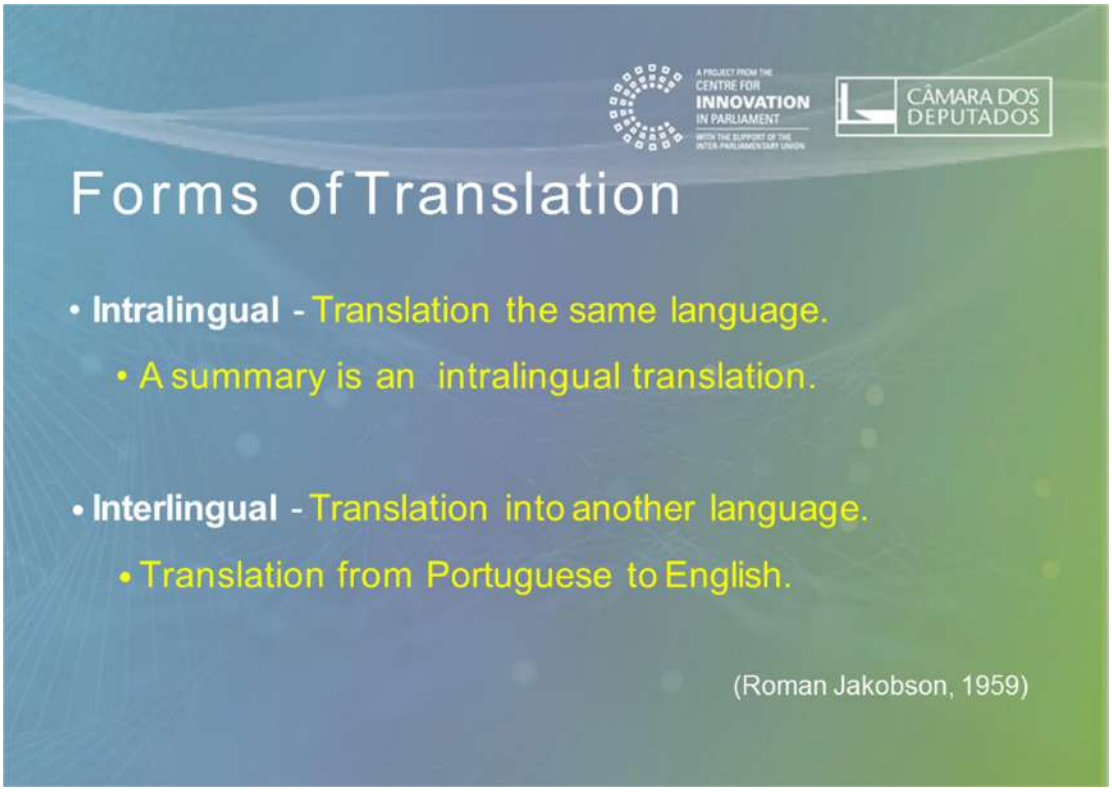
1- The development of na artificial neural machine translation, suitable for legislative documents.

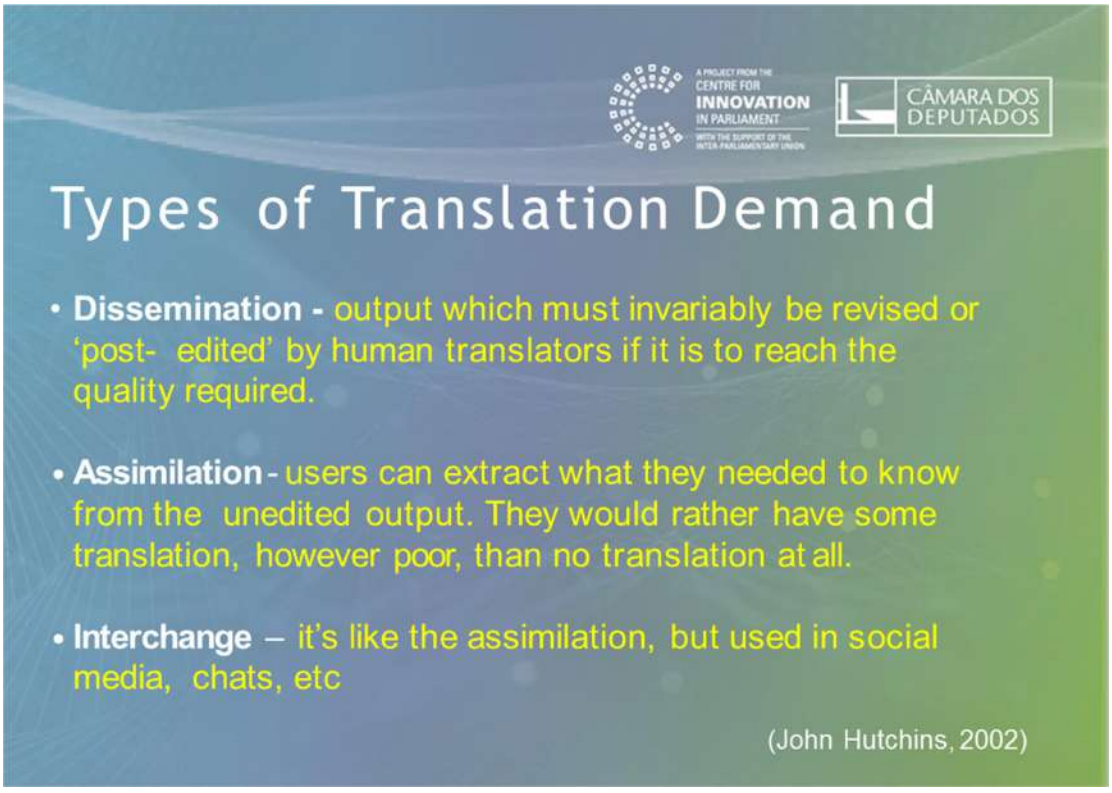
2 – A product that we can share with other parliaments.

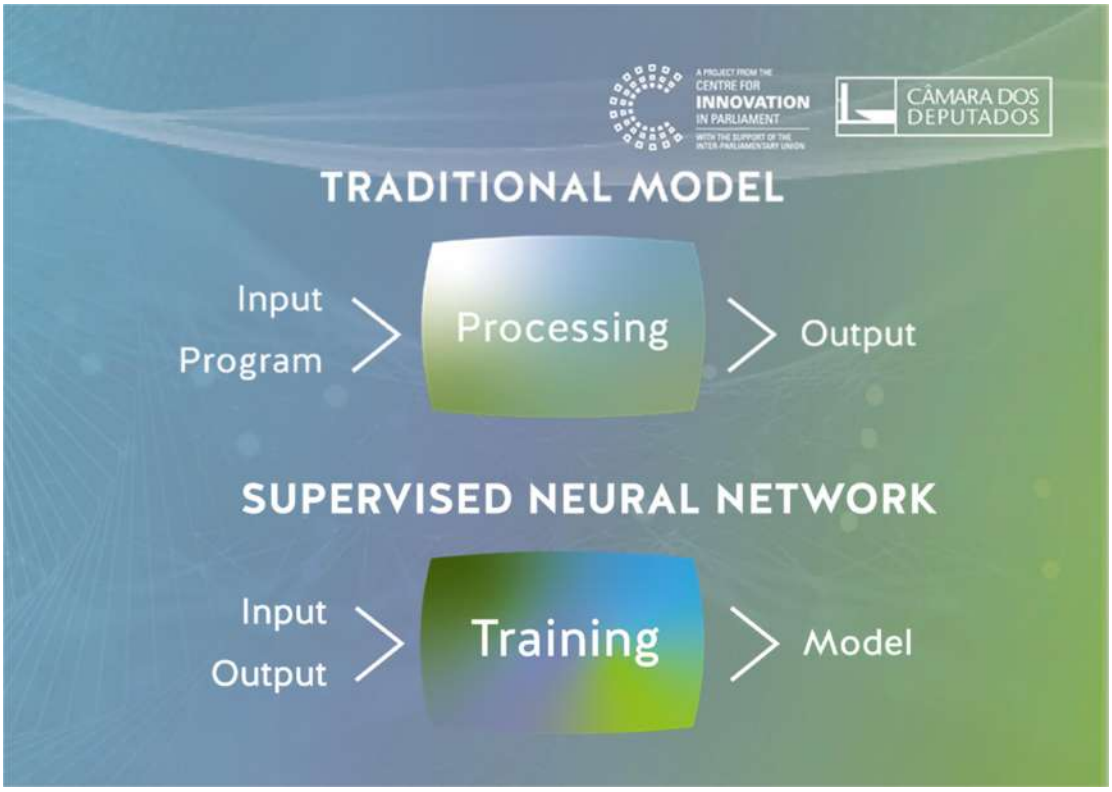
Translation

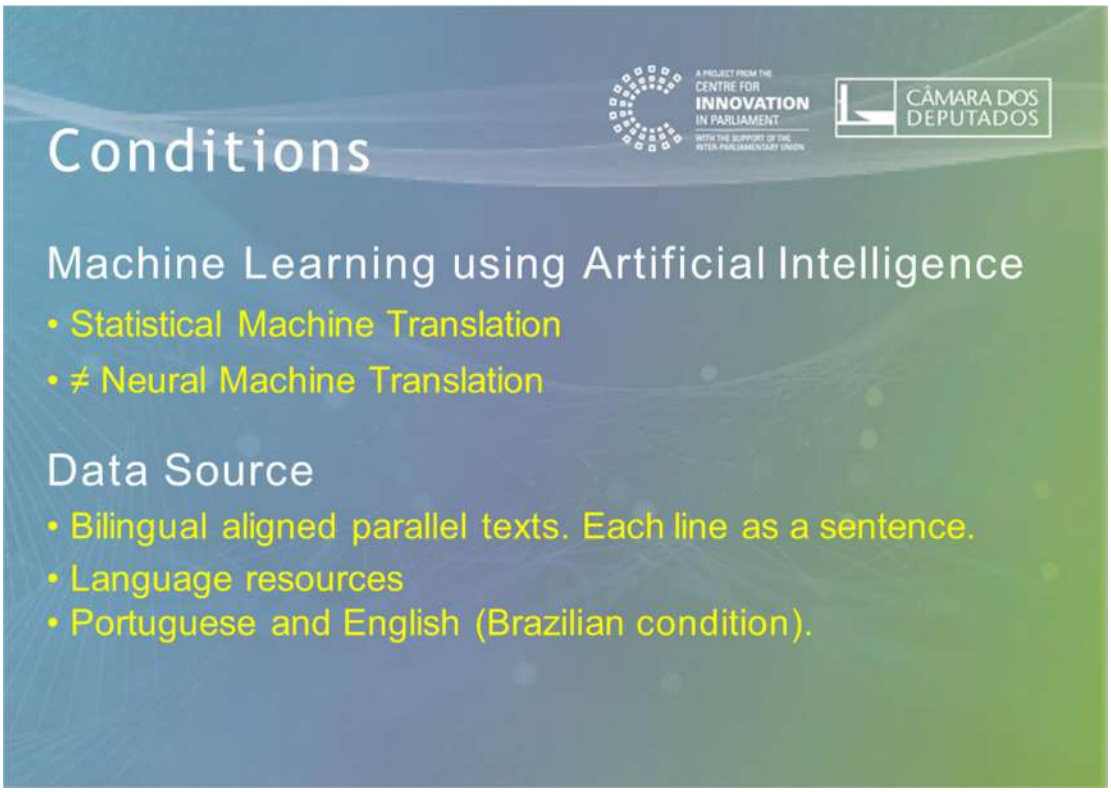
All cognitive experience and its classification is conveyable in any existing language.
Whenever there is deficiency, terminology may be qualified and amplified by loanwords or loan-transl ations, neologisms or semantic shifts, and finally, by circumlocutions.

Roman Jakobson (1959)


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE
INTER-PARLIAMENTARY UNION
CÂMARA DOS
DEPUTADOS
Forms of Translation
• Intralingual - Translation the same language.
• A summary is an intralingual translation.
• Interlingual - Translation into another language.
• Translation from Portuguese to English.
(Roman Jakobson, 1959)


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE
INTER-PARLIAMENTARY UNION
CÂMARA DOS
DEPUTADOS
Types of Translation Demand
• Dissemination - output which must invariably be revised or 'post- edited' by human translators if it is to reach the quality required.
• Assimilation - users can extract what they needed to know from the unedited output. They would rather have some translation, however poor, than no translation at all.
• Interchange – it's like the assimilation, but used in social media, chats, etc
(John Hutchins, 2002)


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
TRADITIONAL MODEL
Input
Program
Processing
Output
SUPERVISED NEURAL NETWORK
Input
Output
Training
Model


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Conditions
Machine Learning using Artificial Intelligence
• Statistical Machine Translation
• ≠ Neural Machine Translation
Data Source
• Bilingual aligned parallel texts. Each line as a sentence.
• Language resources
• Portuguese and English (Brazilian condition).

Conditions

A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

CÂMARA DOS DEPUTADOS

Suitable for Legislative documents

- Legislative sublanguage (protect laws id)
- Vocabulary domain (the dictionary size impacts into memory consumption)

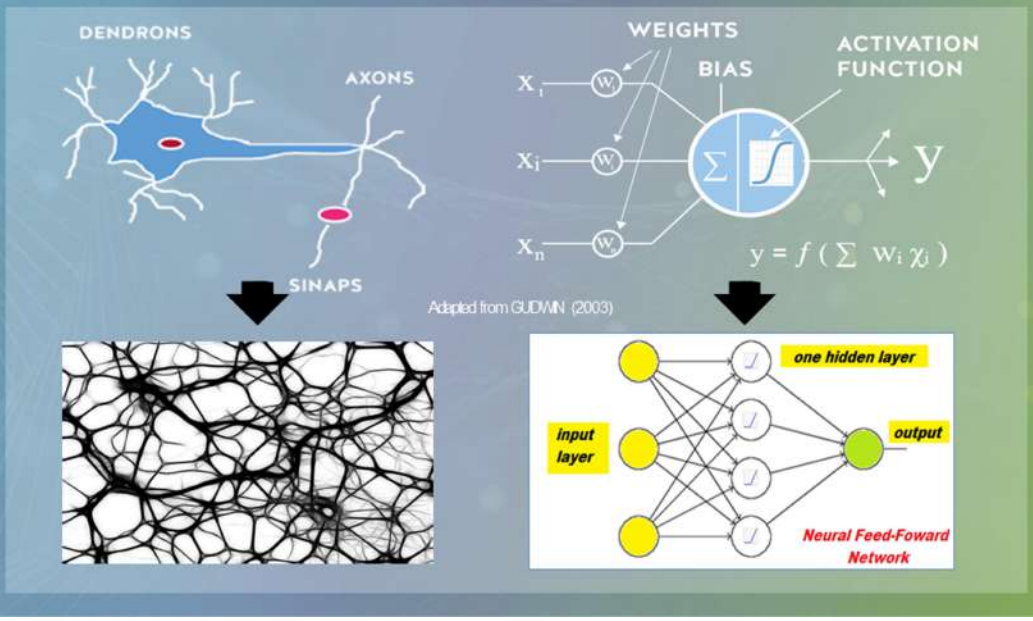
Shareable

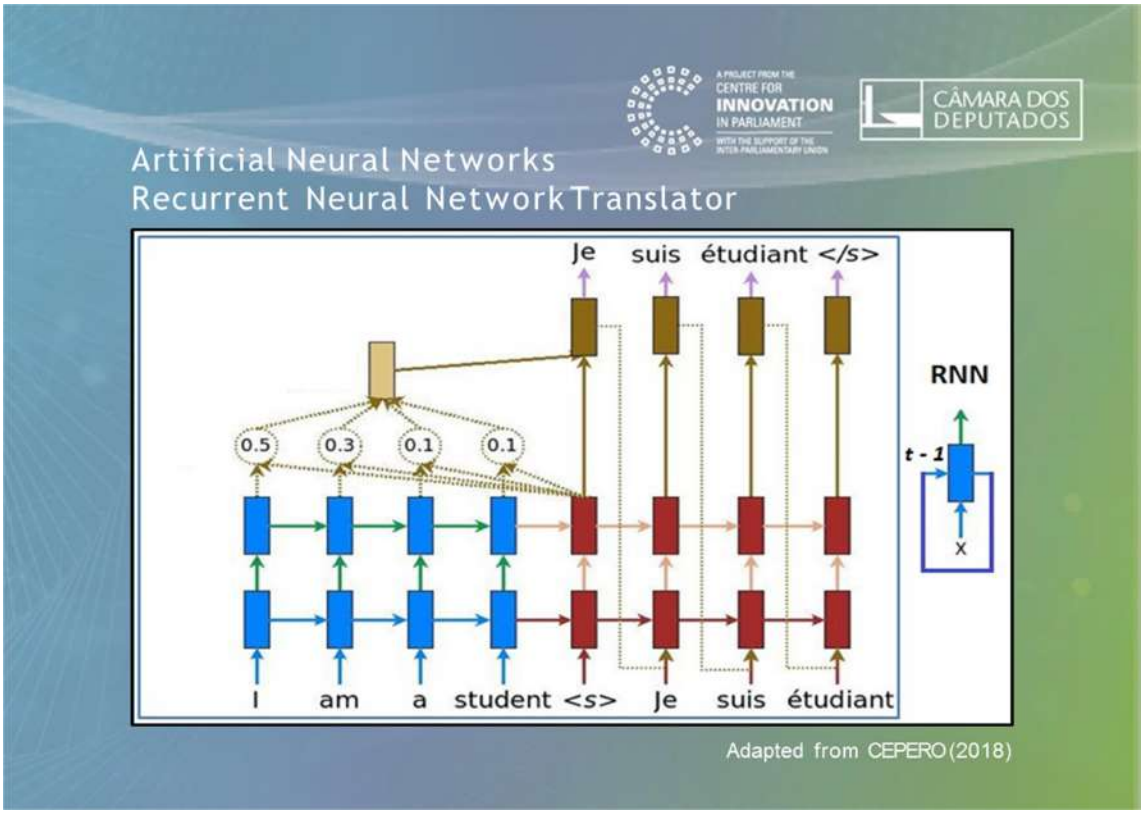
- Started from an Open Source project: OpenNMT
- All fixes and code evolutions remain open source
- Pre- and post-processing programs will be shared free of charge in the inter-parliamentary cloud.

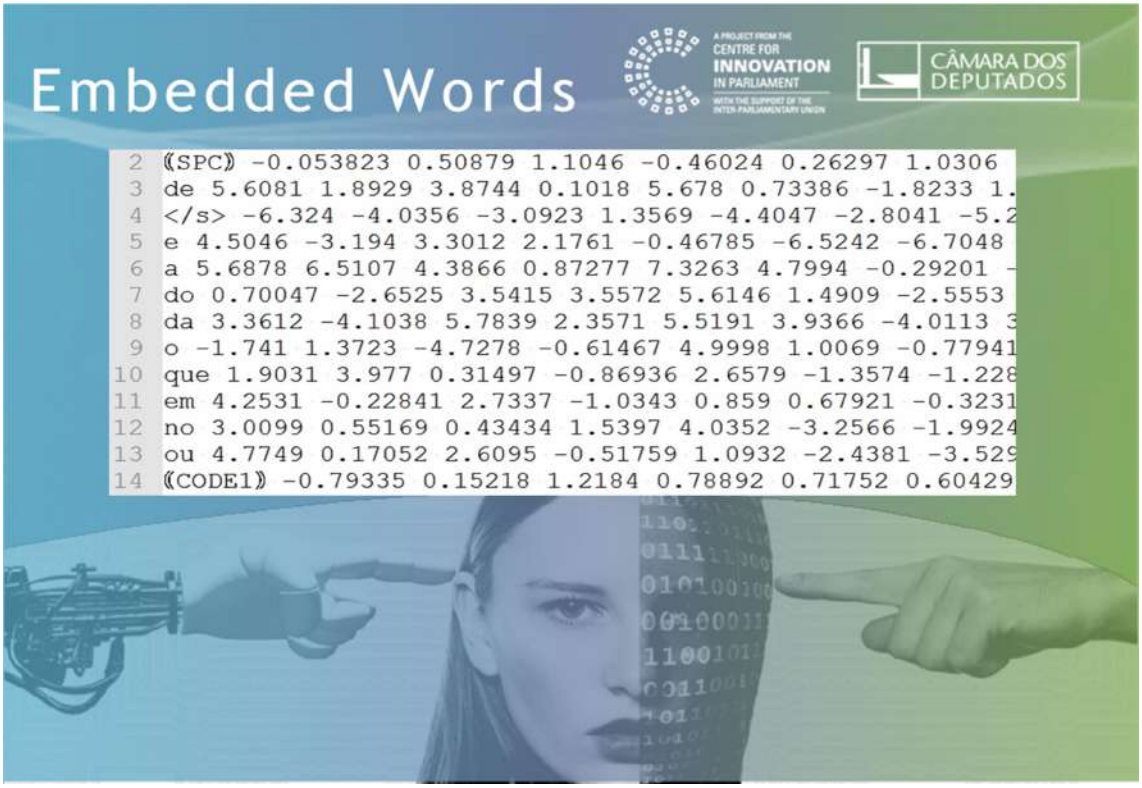
Artificial Neural Networks
The biological inspiration

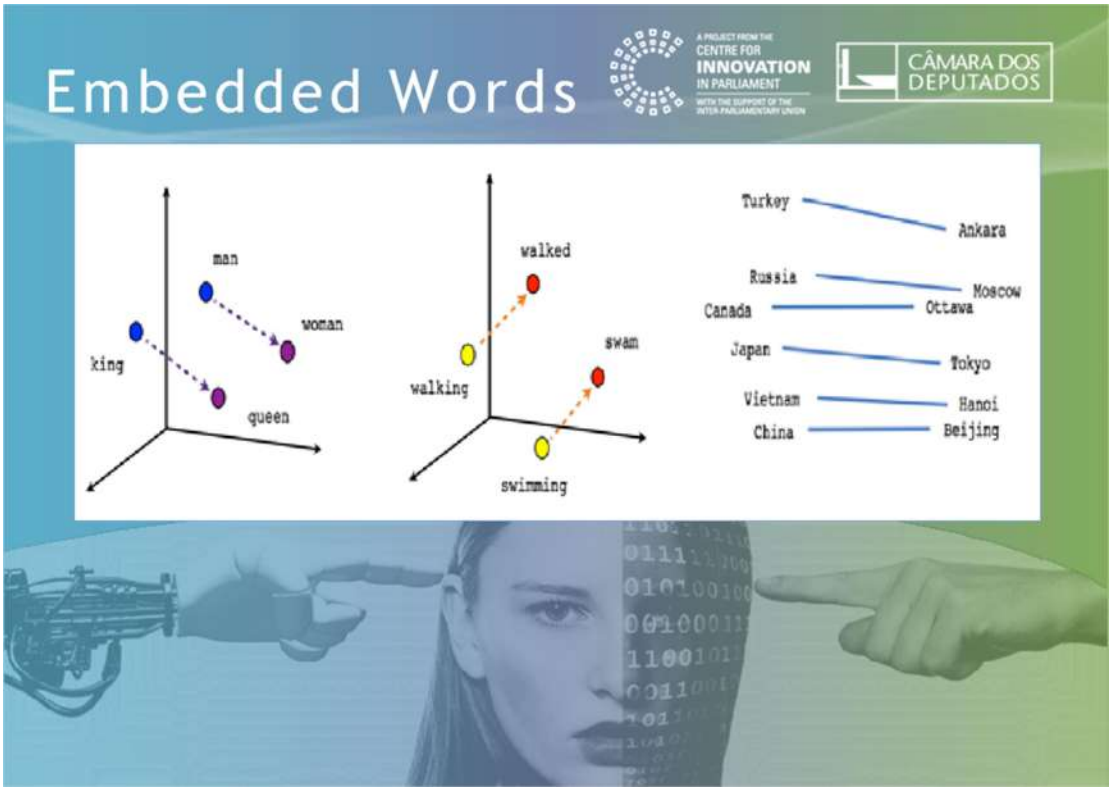
A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

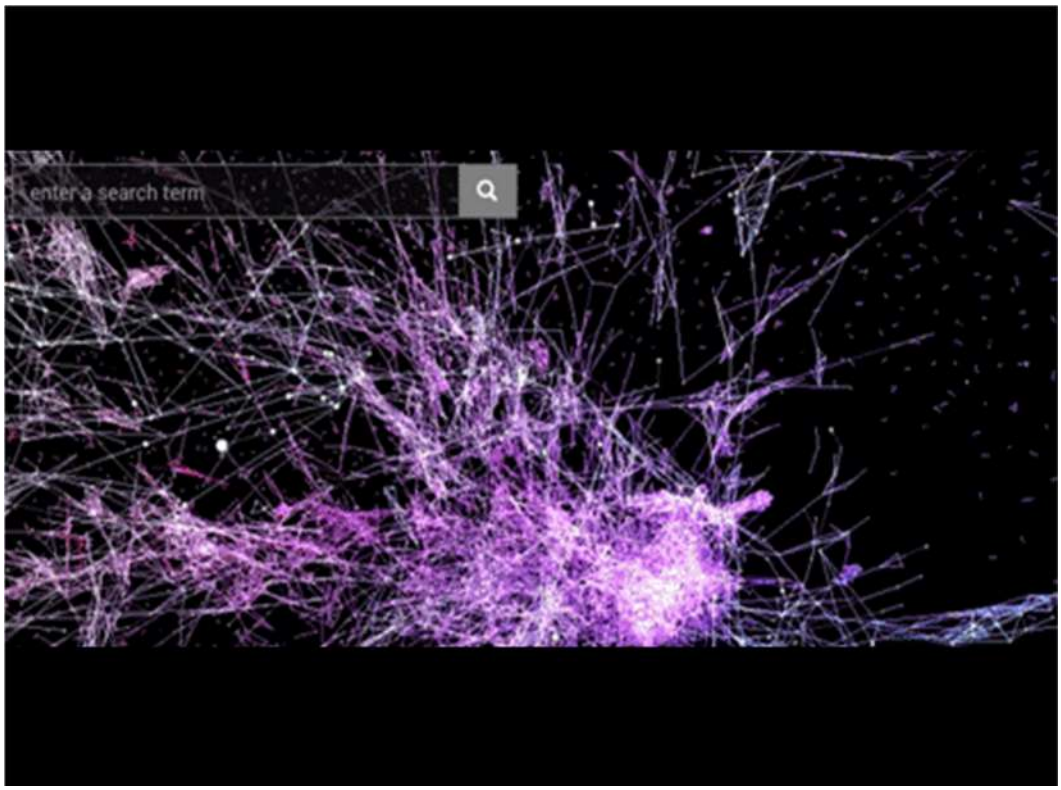
CÂMARA DOS DEPUTADOS




A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Artificial Neural Networks
Recurrent Neural Network Translator
Je
suis
étudiant </s>
RNN
t - 1
X
0.5
0.3
0.1
0.1
I
am
a
student <s>
Je
suis
étudiant
Adapted from CEPERO (2018)


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Embedded Words
2 《SPC》 -0.053823 0.50879 1.1046 -0.46024 0.26297 1.0306
3 de 5.6081 1.8929 3.8744 0.1018 5.678 0.73386 -1.8233 1.
4 </s> -6.324 -4.0356 -3.0923 1.3569 -4.4047 -2.8041 -5.2
5 e 4.5046 -3.194 3.3012 2.1761 -0.46785 -6.5242 -6.7048
6 a 5.6878 6.5107 4.3866 0.87277 7.3263 4.7994 -0.29201 -
7 do 0.70047 -2.6525 3.5415 3.5572 5.6146 1.4909 -2.5553
8 da 3.3612 -4.1038 5.7839 2.3571 5.5191 3.9366 -4.0113 3
9 o -1.741 1.3723 -4.7278 -0.61467 4.9998 1.0069 -0.77941
10 que 1.9031 3.977 0.31497 -0.86936 2.6579 -1.3574 -1.228
11 em 4.2531 -0.22841 2.7337 -1.0343 0.859 0.67921 -0.3231
12 no 3.0099 0.55169 0.43434 1.5397 4.0352 -3.2566 -1.9924
13 ou 4.7749 0.17052 2.6095 -0.51759 1.0932 -2.4381 -3.529
14 《CODE1》 -0.79335 0.15218 1.2184 0.78892 0.71752 0.60429

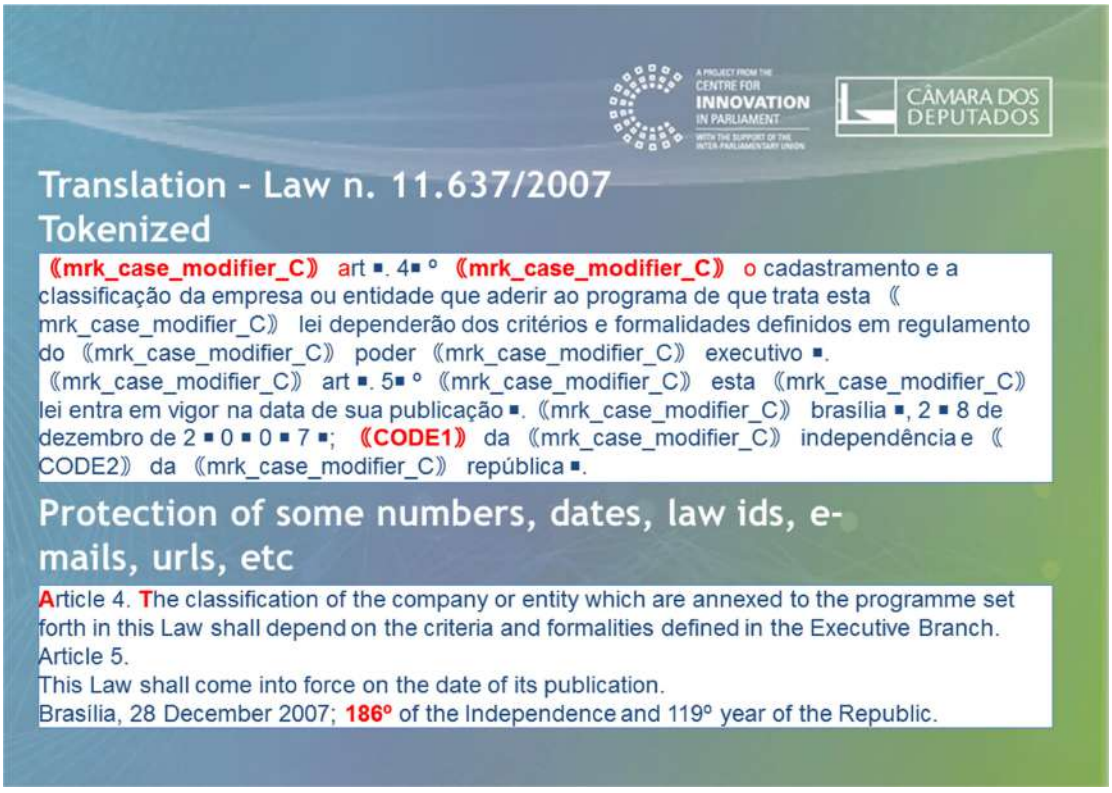




[1]

[1] Slide faz projeção de vídeo baseado no projeto Word2vec Graph, o qual simula a navegação em um espaço de palavras, que está disponível em: https://github.com/anvaka/word2vec-graph.


Translation
A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Law n. 11.637/2007
Source Language: Portuguese
Art. 4º O cadastramento e a classificação da empresa ou entidade que aderir ao programa de que trata esta Lei dependerão dos critérios e formalidades definidos em regulamento do Poder Executivo.
Art. 5º Esta Lei entra em vigor na data de sua publicação.
Brasília, 28 de dezembro de 2007; 186º da Independência e 119º da República.
Protection: numbers, dates, ids, e-mails, urls, etc
Art. 4º O cadastramento e a classificação da empresa ou entidade que aderir ao programa de que trata esta Lei dependerão dos critérios e formalidades definidos em regulamento do Poder Executivo.
Art. 5º Esta Lei entra em vigor na data de sua publicação.
Brasília, 28 de dezembro de 2007; 《CODE1》 da Independência e 《CODE2》 da República


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Translation - Law n. 11.637/2007
Tokenized
《mrk_case_modifier_C》 art ▪. 4▪ º 《mrk_case_modifier_C》 o cadastramento e a classificação da empresa ou entidade que aderir ao programa de que trata esta 《mrk_case_modifier_C》 lei dependerão dos critérios e formalidades definidos em regulamento do 《mrk_case_modifier_C》 poder 《mrk_case_modifier_C》 executivo ▪. 《mrk_case_modifier_C》 art ▪. 5▪ º 《mrk_case_modifier_C》 esta 《mrk_case_modifier_C》 lei entra em vigor na data de sua publicação ▪. 《mrk_case_modifier_C》 brasília ▪, 2 ▪ 8 de dezembro de 2 ▪ 0 ▪ 0 ▪ 7 ▪; 《CODE1》 da 《mrk_case_modifier_C》 independência e 《CODE2》 da 《mrk_case_modifier_C》 república ▪.
Protection of some numbers, dates, law ids, e-mails, urls, etc
Article 4. The classification of the company or entity which are annexed to the programme set forth in this Law shall depend on the criteria and formalities defined in the Executive Branch.
Article 5.
This Law shall come into force on the date of its publication.
Brasília, 28 December 2007; 186º of the Independence and 119º year of the Republic.

A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

CÂMARA DOS DEPUTADOS

Interparliamentary Translations and Definitions (FTAA, Chapter XVIII, Article 1.Definitions)

medida significa qualquer lei, regulamento, procedimento, exigência, orientação ou prática administrativa;

measure means any law, regulation, procedure, requirement, guidance or practice;

measure means any law, regulation, procedure, requirement, administrative guidance or practice;

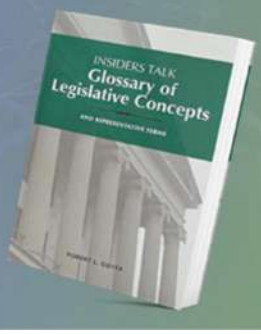
Glossary of Legislative Terms

A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

CÂMARA DOS DEPUTADOS

Assembleia da República - BDTT-AR

proposta de lei	Government bill	projet de loi
projeto de lei	members' bill	proposition de loi
projeto de resolução	draft resolution	proposition de résolution
projeto de revisão constitucional	draft amendments to the Constitution	proposition de révision constitutionnelle



Glossary of Legislative Terms

A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

CÂMARA DOS DEPUTADOS

Brazilian Congress Glossary of Terms
Translation Federal Senate

Lista de conceitos	List of concepts	Liste des concepts	Relación de conceptos
Abertura de Reunião	**Opening of Meeting**	**Ouverture de réunion**	**Apertura de la reunión**
Ato do presidente de comissão que dá início a uma reunião da comissão.	Act of the Commission Chairman who initiates a commission meeting.	Acte du président de commission ouvrant une réunion de celle-ci.	Acto del presidente de la comisión por el cual se abre una reunión de comisión
Ver também:	*See also:*	*Voir également*	*Ver también*
Quórum de Abertura de Reunião *e* Reunião.	Meeting Opening Quorum *and* Meeting.	Quorum d'ouverture de réunion et Réunion.	Quórum de aperture de la reunión y la reunión.
Abertura de Sessão	**Opening of Session**	**Ouverture de séance**	**Apertura de la sesión**
Ato do presidente que declara abertos os trabalhos da sessão plenária.	Act of the President declaring open the proceedings of the plenary session.	Acte du président déclarant ouverte la séance plénière.	Acto del presidente por el que se declaran abiertos los trabajos de la session plenaria.

AUTOMATIC TRANSLATION PT-EN

A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION

CÂMARA DOS DEPUTADOS

8 MILLION
paragraphs from Eurolex

5.3 MILLION
of paragraphs already trained

Training Hit Rate:
96%

ORIGINAL DOCUMENT IN PORTUGUESE

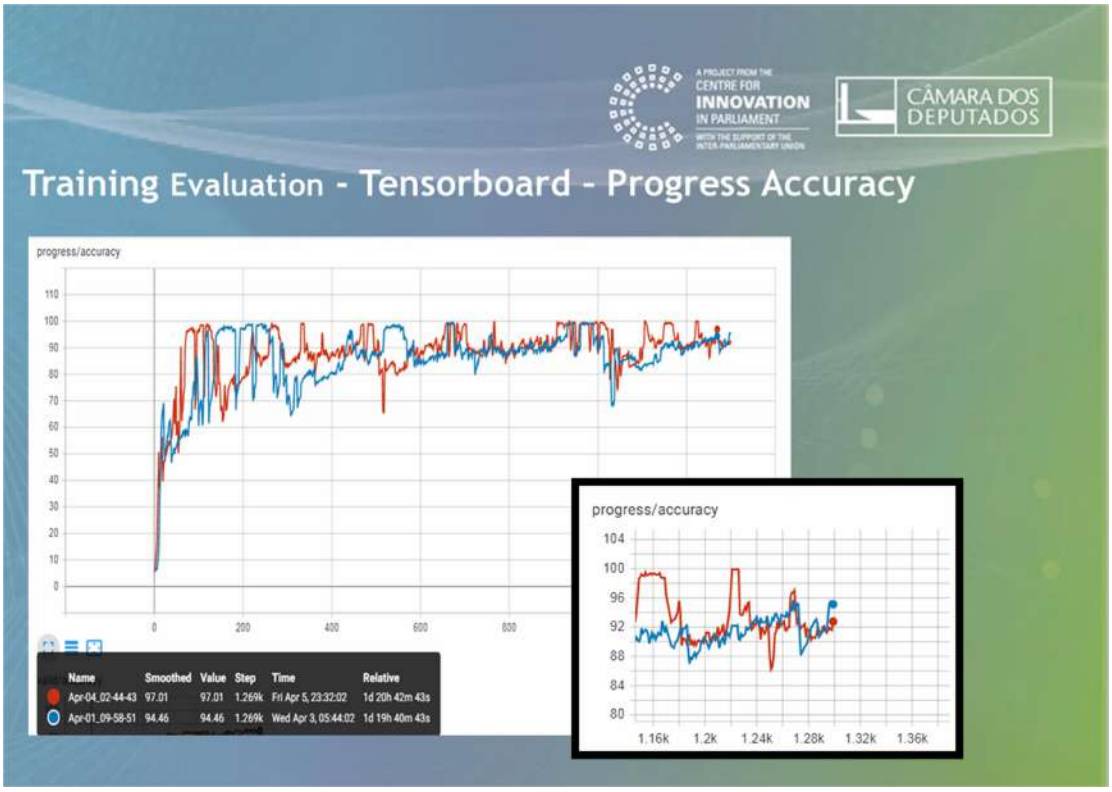
Artigo 20
Notificação voluntária
1 . Sem prejuízo do artigo 3 , as entidades que não tenham sido identificadas como operadores de serviços essenciais e que não sejam prestadores de serviços digitais podem notificar , a título voluntário , os incidentes com impacto importante na continuidade dos serviços por si prestados .

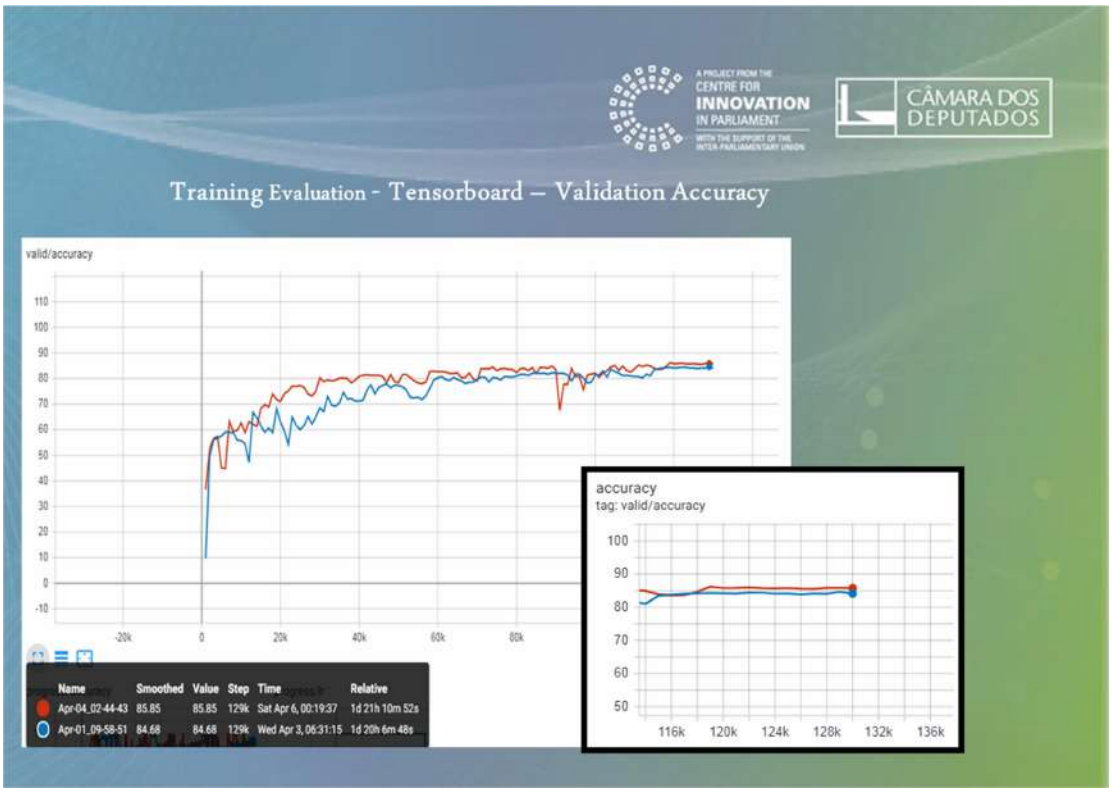
ORIGINAL DOCUMENT IN ENGLISH

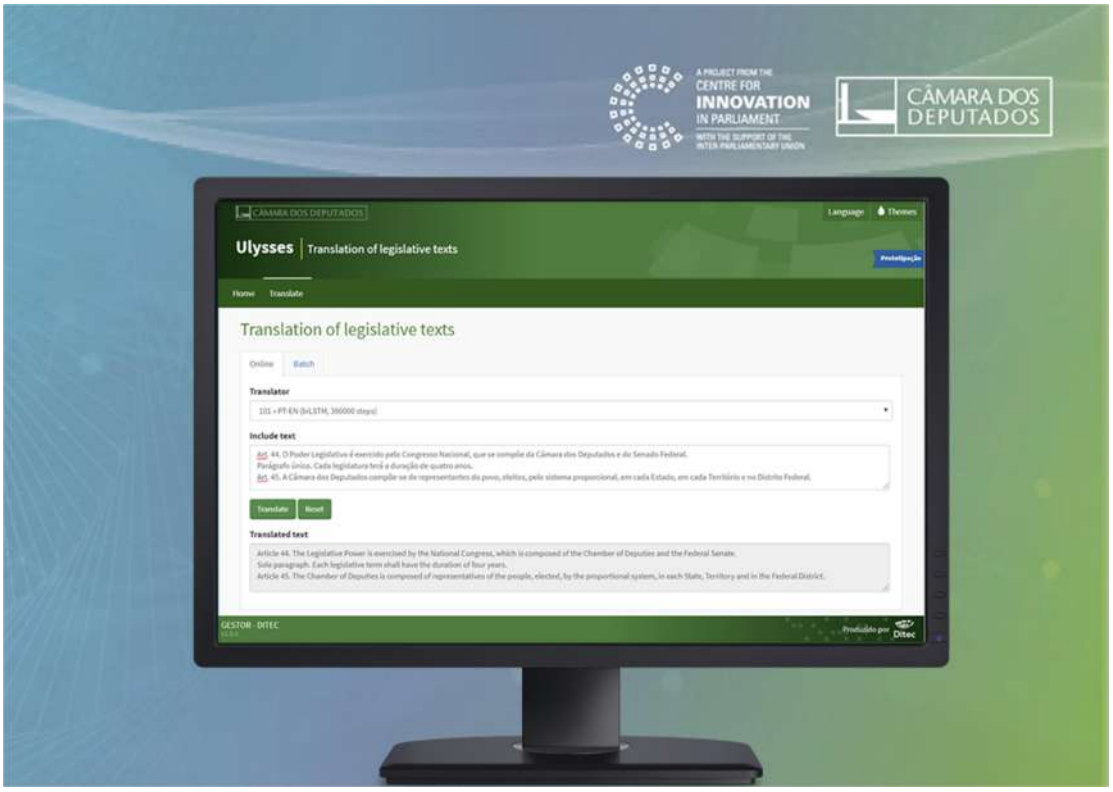
Article 20
Voluntary notification
1 . Without prejudice to Article 3 , entities which have not been identified as operators of essential services and are not digital service providers may notify , on a voluntary basis , incidents having a significant impact on the continuity of the services which they provide .

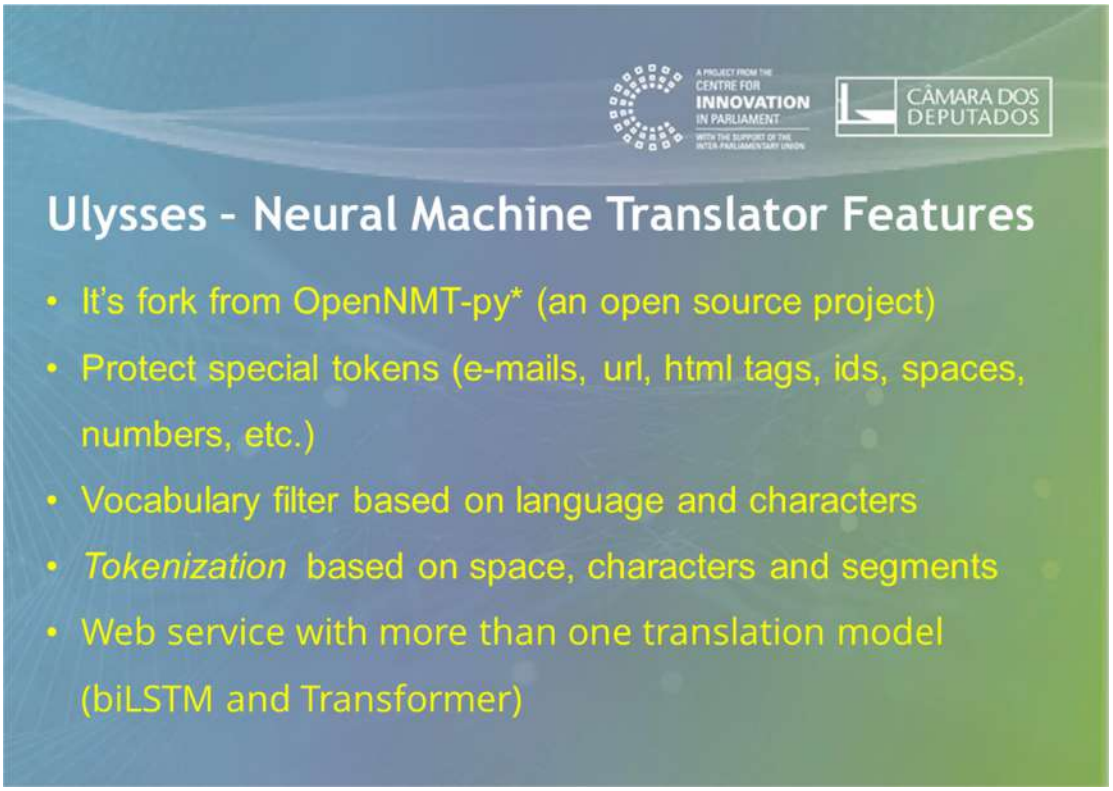
AUTOMATIC TRANSLATION FROM TEXT IN PORTUGUESE*
* 17-Oct-2018

Article 20
Voluntary notification
1 . Without prejudice to Article 3 , entities that have not been identified as operators of essential services and which are not digital service providers may notify , on a voluntary basis , incidents with an important impact on the continuity of services by them .


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Training Evaluation - Tensorboard - Progress Accuracy
progress/accuracy
Name Smoothed Value Step Time Relative
Apr-04_02-44-43 97.01 97.01 1.269k Fri Apr 5, 23:32:02 1d 20h 42m 43s
Apr-01_09-58-51 94.46 94.46 1.269k Wed Apr 3, 05:44:02 1d 19h 40m 43s
progress/accuracy


A PROJECT FROM THE CENTRE FOR INNOVATION IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Training Evaluation - Tensorboard – Validation Accuracy
valid/accuracy
Name Smoothed Value Step Time Relative
Apr-04_02-44-43 85.85 85.85 129k Sat Apr 6, 00:19:37 1d 21h 10m 52s
Apr-01_09-58-51 84.68 84.68 129k Wed Apr 3, 06:31:15 1d 20h 6m 48s
accuracy
tag: valid/accuracy


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE
INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
CÂMARA DOS DEPUTADOS
Language
Themes
Ulysses | Translation of legislative texts
Home
Translate
Translation of legislative texts
Online
Batch
Translator
Include text
Translate
Reset
Translated text
GESTOR - DITEC
Produzido por
Ditec


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE
INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Ulysses - Neural Machine Translator Features
• It's fork from OpenNMT-py* (an open source project)
• Protect special tokens (e-mails, url, html tags, ids, spaces, numbers, etc.)
• Vocabulary filter based on language and characters
• *Tokenization* based on space, characters and segments
• Web service with more than one translation model (biLSTM and Transformer)

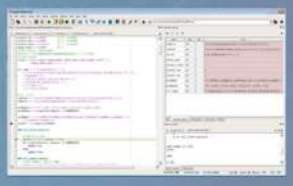
Ulysses - Neural Machine Translator Features

- Compatible with Tensorboard
- Can be adapted to convert from speech-to-text and translate it.
- Can be adapted to summarize as a form of translation
- Being adapted to work with aligned word embbedings

Lab Resources

Software

- OpenNMT
- Moses (language prefixes)
- Spyder (Python)
- Sqlite
- Flask
- Python
- Libraries (Pytorch, Pandas, Pyonmttok, Polyglot, Tensorboard, etc)



Hardware

- Virtual Machine:
  - Ubuntu 16.04
  - 20 GBRAM
  - 32 cores
- 2 Computers:
  - Ubuntu 16.04
  - ~20 GB RAM (Computer)
  - Geforce GPU 1080
    - 8 GB GDDR5X
    - 2560 CUDA cores

Comparing ...

Google's NMT (WNMT 2016)	CD's NMT Lab (IPU - Brazil)
November 2016	November 2018 (GPU) August 2018 (CPU)
8 LSTM hidden layers (1 bi-directional encoder hidden	2 - 3 LSTM hidden layers (1 bi-directional encoder hidden
GPU	GPU (PCIe 3.0 in PCIe 2.0 slot)
Residual Connections & Dropout	Residual Connections & Dropout
8 Tesla k80 (2 x K40) 24GB	1 GTX 1080 8GB
Generic Big Data Datasets Anything	Legislative Documents Datasets Laws & Legislative Bills Formatting Parliamentary Glossary of Terms

"IBM Cloud: VM with Tesla V100, Nvidia + TPU --- $ 2,300/month"
"Microsoft cloud: VM with Tesla V100, Nvidia + TPU --- $ 1,800/month"
"Desktop: Titan RTX 24G, 4608 cores + TPU --- $ 2,500"
"Workstation: Quadro RTX 6000, 32G 4608 cores + TPU --- $ 4,700"

Lessons Learned

1. Translation is a complex system. Better results using only one variance of language.
2. CPU only to preprocess data. Use graphics processing units (GP or TPU) for training and translation (about 10 times faster).
3. Each parliament, in addition to providing its glossary of legislative terms in English, may endeavor to include two more languages.
4. The great value of human translators.
5. Preprocessing text has a relevant impact for the research and translation. Than, we can do it better.


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
Starting to ...
• Translate all Brazilian laws and legislative bills
• Translate from English to Portuguese and variants
• Use summarized translation (intra & inter)
• Validate an new approaches that will improve the translation quality
• Mutilingual translation through computational interlingua
• Train the translator without bilingual parallel texts


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE INTER-PARLIAMENTARY UNION
CÂMARA DOS DEPUTADOS
References
• Freitas, Eduardo. The artificial intelligence in the automatica translation of legislative documents: An application in the parliamentary research, 2019.
• Klein, Guillaume et al. OpenNMT: Open-Source Toolkit for Neural Machine Translation, 2017. SYSTRAN & Harvard University.
• Hutchins, John. Machine translation today and tomorrow, 2002.
• Jacobson, Roman. On linguistic aspects of translation, 1959.
• Mikolov, Tomas et al. Advances in neural information processing systems, 2013
• Paul, Apostle. The First Epistle to the Corinthians, 54-57 AD.


A PROJECT FROM THE
CENTRE FOR
INNOVATION
IN PARLIAMENT
WITH THE SUPPORT OF THE
INTER-PARLIAMENTARY UNION
CÂMARA DOS
DEPUTADOS
THANK YOU
Eduardo A Mello Freitas
IT Solution Manager and Consultant
eduardo.freitas@camara.leg.br

ANEXO A - INTER-PARLIAMENTARY OPEN DATA CLOUD

Inter-Parliamentary Union

Inter-Parliamentary Open Data Cloud

A project within the Centre for Innovation in Parliament

CÂMARA DOS DEPUTADOS

CONCEPT NOTE

27 February 2019

Project objective

The goal is to enhance parliamentary effectiveness and efficiency within the lawmaking process, by building a single point of access to legislative information from multiple countries in multiple languages. To achieve this goal, the project intends to aggregate open data published by parliaments around the world and to apply machine learning techniques to automate translation of this open data.

Rationale

Starting a new bill is a process that involves a lot of research, studies and hearings, which often requires significant time, effort and costs. One of the first steps in the law-making process is usually to consider what laws have already been adopted on the same topic in other countries.

Currently there are many barriers to finding legislative documents from other countries. Many countries publish bills and laws, but each country's data is accessible in a different place and in a different way. These documents are typically only published in the national language, and not translated into other languages. The sheer volume of data can be overwhelming, and make it difficult to locate the relevant documents.

Providing a central point in a single language for parliaments' open data will facilitate the work of MPs, their assistants, students, researchers and anyone interested in analyzing legislative issues such as bills and laws.

Any MP wishing to draft a bill could first check the laws of others countries on the same subject. He/She could build on the information and knowledge generated by other parliaments (research, studies, debates, …) instead of starting from the scratch. This would improve the results with less effort, which would provide better efficiency at less cost.

Outputs

The main project output is a Distributed Open Data Platform that can be accessed by everyone as if all the data were in the same place. In practice, each parliament will continue to host, manage and own its open data according to a unified pattern, but the platform will offer access to data from a multitude of parliaments in order to render better studies.

Machine learning techniques will be used to create automated translation of bills and laws from one language to another, so legislation from multiple countries can be viewed in a single language.

Dashboards will be built to enable users to get quick and easy access to the topics in which they are interested.

The first phase (with the "G1 Parliaments") will see the creation of this "inter-parliamentary cloud". Subsequently, dissemination of the open data model will begin. This will make use of distance learning and the regional hubs within the Centre for Innovation in Parliament to transfer knowledge on open data management to those Parliaments that don't yet have this expertise.

Expected benefits

- Simplify access to legislative data from a large number of parliaments;
- Promote deeper understanding of legislation in a global scale;
- Promote the production of analyses and comparative studies among parliaments;

- Identify opportunities for improvements in parliaments' legislative work and their relations with citizens;
- Identify correlations between the legislation characteristics and the public policies applied in each country (for example, laws X PISA results);
- Promote a knowledge network among parliaments that participate in the project on open data platforms, open data governance and data science;
- Share information among parliamentary specialists;
- Increase parliaments' transparency globally.

Participants

The Inter-Parliamentary Open Data Cloud is a collaborative project involving a group of parliaments that publish open data, coordinated by the Brazilian Chamber of Deputies. It will function as one of the hubs within the IPU's Centre for Innovation in Parliament.

Participants will collectively define the Open Data Plan and Governance Model for the project. In the initial phase, the participating parliaments will be those that are already producing documents in open data formats. These parliaments will be known as the "G1" group.

Participants are expected to be parliamentary staff who have technical mastery of open data and/or artificial intelligence, in addition to being familiar with the legislative process in parliaments.

Expected workload

Participating parliaments should expect to have to do some work to contribute to the Inter-Parliamentary Open Data Cloud. Much of the work will be done online. Occasional face-to-face meetings of parliamentary experts will be organized at key milestones in the project. The working language for the project will be English.

Expected roles of participating parliaments include:

- Actively review and comment on the Open Data Plan
- Define with the group the Patterns, Standards and data formats
- Define with the group the datasets that will comprise the cloud
- Establish the dissemination strategy
- Provide the data in the country's national language for the automated text translation
- Refine the translation model
- Generate its own open data for the cloud
- Deployment, operation and support for its own open data

More information

Documents

Presentation of the Inter-Parliamentary Cloud Open Data project at the World e-Parliament Conference, 5 December 2018 https://www.ipu.org/download/6132

Centre for Innovation in Parliament https://www.ipu.org/node/9774

Contacts

Patricia Gomes Rêgo de Almeida, Brazilian Chamber of Deputies patricia.almeida@camara.leg.br

Rodolfo Cezar Ranulfo Vaz, Brazilian Chamber of Deputies rodolfo.vaz@camara.leg.br

Andy Richardson and Avinash Bikha, IPU innovation@ipu.org

ANEXO B – LINHA DO TEMPO

2018	27/Março	Reunião sobre a Nuvem Interparlamentar. Proposta de uso da tradução automática na tradução.
	30/Maio	Reunião sobre aplicação das Nações Unidas. Proposta de adaptação da API do eTranslation da União Europeia para a Câmara dos Deputados.
	Junho	Extração de dados do Eur-Lex em Português, Inglês, Francês, Espanhol e Italiano. Estudo sobre melhor forma de filtrar e manter os dados extraídos. Pesquisa de projetos abertos que possam servir de base a um tradutor, que tenham obtido bons resultados em workshops e que possam ser aproveitados em projeto em curso de Reconhecimento de Orador para Plenários e Comissões.
	16-21/Junho	Participação, em Bruxelas, de reuniões com grupos da União Interparlamentar, do Parlamento Europeu e de apresentação do Instituto de Tecnologia de Karlshure.
	Julho	Estudo aprofundado dos projetos Word2Vec e OpenNMT. Início dos testes em CPU e especificação de placas GPU para aquisição.
	Agosto	Testes do OpenNMT-py com tokenização pelos utilitários em Lua e Perl. Aproveitamento de arquivos referentes à língua portuguesa do projeto Moses.
	Setembro	Acesso remoto ao computador à Cârmara dos Deputados. Estudo e codificação para aprimoramento do vocabulário, com conversão de PT-PT para PT-BR, e uso de diferentes tipos de tokenização.
	Outubro	Tradução automática para a pesquisa interparlamentar substitui projeto anterior. Início do novo projeto para apresentação em disciplina do Mestrado. Estudo para reduzir a presença de palavras desconhecidas o mínimo de aumento no tamanho do vocabulário.
	Novembro	Recebimento e instalação de duas placas GPU modelo GTX 1080 de 8G. Mudança no modo de acesso à rede da Câmara dos Deputados a fim de aumentar a estabilidade da rede e mantê-la por mais tempo.
2019	Janeiro	Revisão de parâmetros para uso com GPU e criação de novos parâmetros. Alteração das rotinas de pré-processamento do projeto OpenNMT-py. Revisão de conceitos de estruturas morfológicas das palavras.
	Fevereiro	Compatibilização com a versão 0.7.1 do Pytorch. Primeira versão da rotina de proteção de termos. Interrupção de treinamento para revisão de parâmetros.
	13/Março	Qualificação aprovada.
	Março	Correção de rotina de tratamento de erros no PyTorch. Correção do serviço de tradução automática na Web. Conclusão das rotinas de proteção incluindo espaços e caracteres repetidos.
	Abril	Correções e ajustes com a versão PyTorch 1.0 mais estável. Uso de propriedades junto com as palavras não funciona mais no OpenNMT-py. Melhorias no modelo Transformer dependem de mais memória. Configuração da rede neural LSTM com melhor resultados nos testes. Página do tradutor em pleno funcionamento.

2019	Maio	Código do serviço estava incompleto no tratamento de salto de linha. Treinamento prolongado levou ajustes durante o treinamento a resultados muito bons.
	27-31/Maio	Apresentação do tradutor automático em evento da Nuvem União Interparlamentar. Solução surpreendeu por conta dos poucos recursos de hardware.
	Junho	Inclusão dos apêndices e da apresentação após revisão do relatório.
	4/Julho	Defesa
	Julho –Agosto	Revisão Final para Homologação