INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA

Tradução para língua de sinais

DIOGO TOMÁS RAMOS DOS SANTOS OLIVEIRA
Outubro de 2017

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Tradução para língua de sinais

Diogo Tomás Ramos dos Santos Oliveira

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas de Informação e Conhecimento

Orientador: Nuno Filipe Fonseca Vasconcelos Escudeiro

Co-orientador: Paula Maria de Sá Oliveira Escudeiro

Porto, outubro 2017

Página em branco

Dedicatória

Dedico este trabalho aos meus pais, Tomás Oliveira e Maria Oliveira.

Página em branco

Resumo

As pessoas surdas sofrem de exclusão social e de informação devido a dificuldades de comunicação com pessoas não-surdas. Igualdade e inclusão serão promovidas se existiram soluções que suportem uma comunicação mais efetiva entre a comunidade surda e não-surda. Esperamos ser capazes de disponibilizar traduções precisas entre língua natural escrita e língua gestual, através um processo sustentável e escalável para facilitar a comunicação entre pessoas surdas e não-surdas no dia-a-dia.

Palavras-chave: Surdez, Língua Gestual Escrita, Tradução, Corpora Paralelo, Deep Neural Networks, Neural Machine Translation.

Página em branco

Abstract

Deaf people suffer from info and social exclusion due to difficulties in communicating with non-deaf people. Equity and inclusion will be promoted if there are solutions assisting a more effective communication between deaf and non-deaf communities. We expect to provide accurate translations from written to sign languages through a sustainable and scalable process to facilitate the communication between deaf and non-deaf people in daily life.

Keywords: Deaf, Sign Writing, Translation, Parallel Corpora, Deep neural networks, Neural Machine Translation.

Página em branco

Agradecimentos

Ao meu orientador o Prof. Nuno Filipe Fonseca Vasconcelos Escudeiro (Docente no Instituto Superior de Engenharia do Porto) pela paciência, dedicação, incentivo e por todo o valiosíssimo apoio que deu ao longo do presente estudo.

Ao ISEP – Instituto Superior de Engenharia do Porto por permitir as condições necessárias à realização deste trabalho.

Aos meus pais, um agradecimento especial por todo o esforço feito.

Um obrigado especial à minha namorada, Emilia Skowrońska. A ela agradeço toda a compreensão, tempo, carinho e motivação dados incondicionalmente nos momentos mais complicados.

Não posso deixar de agradecer também ao meu amigo Eng.º Diogo Silva pela motivação e conhecimentos partilhados.

Página em branco

Índice

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

Lista de Figuras

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

Lista de Tabelas

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

Acrónimos e Símbolos

Lista de Acrónimos

IA	Inteligência Artificial
LGP	Língua Gestual Portuguesa
ASL	American Sign Language

Lista de Símbolos

Inserir página em branco apenas se necessário de modo a que a próxima secção comece numa página à direita

1 Interpretar o problema

1.1 Contexto

Segundo a Associação de Surdos do Porto, existem em Portugal cerca de 100.000 pessoas com problemas relacionados com surdez. A Associação de Surdos da Alta Estremadura estima que cerca de 33.000 pessoas [1] utilizam a Língua Gestual Portuguesa (LGP) como forma de comunicação. De forma a tornar possível a inclusão destas pessoas na vida ativa, é necessário criar conteúdos traduzidos da Língua Portuguesa para LGP. Para tal, é necessário contratar especialistas que tenham conhecimento na área da tradução. Este é um processo demorado e caro e que raramente é aplicado pela maioria das empresas.

1.2 Problema

A produção de conteúdos digitais (educativos e outros) para surdos requer a participação de especialistas em produção e edição de vídeo e intérpretes de língua gestual. É um processo demorado e oneroso dada a quantidade de recursos especializados de que necessita. Estas circunstâncias dificultam a inclusão de surdos na vida ativa e na sociedade de uma forma geral.

1.3 Objetivo

Criação de um sistema automático de tradução entre texto e língua gestual que permita produzir recursos digitais próprios para surdos com rapidez e a baixo custo, contribuindo para promover a inclusão e igualdade de oportunidades da comunidade surda internacional. Facilitar a produção de conteúdos em língua gestual, tornando-os mais baratos de produzir e massificando a sua disponibilidade.

1.4 Resultados esperados

Este sistema recorrerá a técnicas de tradução de corpora paralelos e a aprendizagem automática para permitir a tradução automática ou semiautomática de qualquer língua oral para a respetiva língua gestual.

1.5 Análise de valor

Através do modelo *New Concept Development* (ver secção 2.2.1), é possível verificar que o sistema a desenvolver é considerado uma oportunidade. Esta surge assim que se verifica que pessoas com surdez têm dificuldade em comunicar com pessoas sem problemas auditivos. É possível verificar uma certa exclusão social deste grupo de pessoas.

No que respeita a valor para a respetiva comunidade, é possível antecipar uma melhoria na comunicação e nos conteúdos educacionais disponíveis. Irão beneficiar de uma maior inclusão social com a expansão da cobertura da tecnologia.

Após esta análise, foi criado um modelo *Canvas* para o projeto (ver Figura 5). O método AHP (ver secção 2.2.2) auxiliou à escolha das melhores alternativas tecnológicas a utilizar.

1.6 Abordagem preconizada

O website *Signwriting.com* possui diversos *corpora* em diversas línguas naturais e as suas respetivas traduções para a língua gestual escrita. Estes *corpora* vêm sendo atualizados ao longo dos últimos 20 anos [2] e compreendem enorme conjunto de dados por explorar. Será possível, através da aplicação de técnicas de inteligência artificial, a criação de um sistema de tradução automática, usando os dados contidos nos *corpora* obtidos.

O objetivo principal deste projeto é fornecer às pessoas com surdez um sistema de tradução automática para ser possível traduzir noticias, livros, websites e permitir simples interações com pessoas sem problema de surdez.

Pretende-se, ainda, disponibilizar um *avatar* 3D (ver secção 3.1.5) que permita facilitar ao utilizador a leitura dos sinais obtidos a partir da tradução.

1.7 Estrutura da dissertação

Esta dissertação encontra-se dividida em seis capítulos, que, por sua vez, se encontram subdivididos.

O capítulo inicial propõe-se a introduzir a dissertação numa perspetiva global da problemática. De seguida, e de uma forma concisa, é apresentado o contexto, o problema e os objetivos. Por fim, uma breve análise de valor.

O segundo capítulo é usado para descrever a problemática com um maior grau de detalhe. De forma igualmente mais detalhada, é apresentada a análise de valor e o estado da arte desta dissertação.

O terceiro capítulo tem o objetivo de relacionar problema, solução e tecnologias disponíveis.

O quarto capítulo apresenta pormenores sobre os corpora utilizados neste projeto. Apresenta detalhes sobre a arquitetura do projeto e o seu funcionamento.

O quinto capitulo fala da avaliação do sistema implementado. Começa por descrever as métricas a usar. Por fim, são apresentados os resultados desta dissertação.

O sexto capítulo é responsável por apresentar as conclusões da dissertação, contribuições e trabalho futuro neste projeto.

Inserir página em branco apenas se necessário de modo a que o próximo capítulo comece numa página à direita

2 Contexto e Estado da arte

2.1 Detalhes sobre contexto e problema

2.1.1 Processos e intervenientes

2.1.1.1 Associações de Surdos

Estas associações ajudam a facilitar a vida da pessoa surda no seu dia-a-dia. Esta ajuda pode ser no sentido da tradução de livros e noticias ou simplesmente ensinar língua gestual para que a pessoa se possa expressar.

2.1.1.2 Pessoa surda

A pessoa surda, ao contrário do que a maioria das pessoas pensa, não é capaz de ler língua natural escrita. Visto que a sua capacidade auditiva é reduzida ou inexistente, a pessoa surda é incapaz de interiorizar os sons que caracterizam cada palavra, tornando impossível a sua leitura e com isto, tornando virtualmente impossível qualquer interação com a população em geral.

2.1.2 Restrições existentes

A principal restrição existente à criação deste projeto é a virtual inexistência de corpora que incluam língua gestual escrita. Visto que o número de pessoas a utilizar esta forma de comunicação é limitado, é quase impossível encontrar corpora paralelos que inclua esta língua. Existem corpora paralelos entre praticamente todas as línguas naturais [3]. A pouca investigação feita na área da língua gestual escrita é, maioritariamente feita com recurso a pequenos corpora na língua gestual escrita do país onde se está a proceder à investigação. Ou então são usados vídeos com uma pessoa a expressar-se em língua gestual local.

Outra restrição existente, mas esta relacionada com a criação de conteúdos traduzidos, é o elevado custo associado à criação de recursos digitais para surdos, pois é necessário contratar especialistas.

2.2 Análise de valor

2.2.1 The new concept development model

Segundo Peter Koen, este modelo é constituído por três partes (ver Figura 1):

- O *engine,* que representa a estratégia empresarial da organização. A organização tem uma determinada visão, dependendo da sua cultura interna.

- Os cinco elementos do modelo: a identificação da oportunidade (*opportunity identification*), a análise da oportunidade (*opportunity analysis*), a geração da ideia (*idea genesis*), a seleção da ideia (*idea selection*) e a desenvolvimento do conceito e tecnologia (*concept & technology development*).

- Os fatores influenciadores, são fatores que a organização não controla poderão influenciar o processo até à comercialização. Estes fatores vão desde o cliente até à concorrência, passando pelo desenvolvimento tecnológico do mundo.

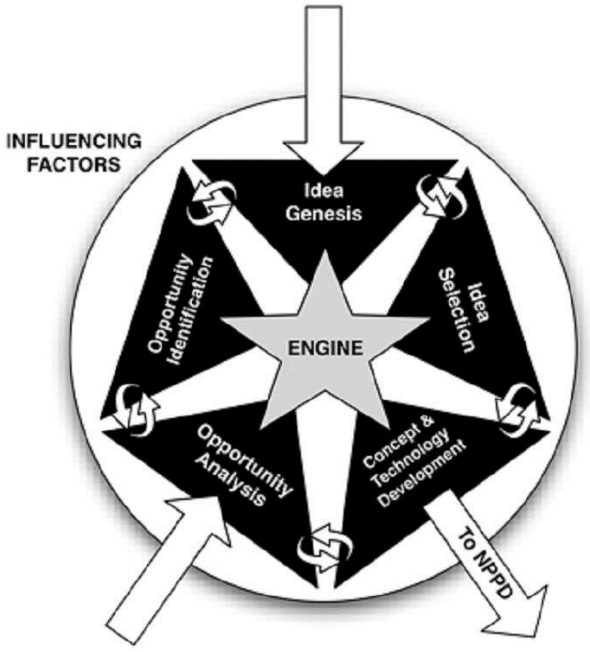


Figura 1 - Modelo de desenvolvimento de novo conceito.

Como o modelo apresenta uma forma circular, é de esperar que os elementos que o constituem interajam, independentemente da ordem. É igualmente expectável que cada elemento seja usado mais que uma vez.

Na Figura 1 é possível notar que o modelo é composto por setas que apontam tanto para fora como para dentro do modelo. Estas são representativas da fase inicial do modelo e a fase final de conceção do conceito para o novo desenvolvimento do produto[1] (NPD) ou para o processo tecnologia por etapas[2] (TSG), respetivamente.

Passou-se então para a fase de experimentação com o modelo referido.

Ideia 1: Implementação de algoritmos tradicionais de Inteligência artificial para criar um modelo de tradução.

Análise à ideia 1: seria uma ideia interessante, mas existem técnicas mais recentes e com um maior nível de sucesso na criação de modelos de tradução.

Ideia 2: Implementação de algoritmos de *Deep Learning* para a criação de um modelo de tradução.

Análise à ideia 2: esta é a ideia escolhida e a mais consistente.

2.2.2 Método AHP

Para fazer uma análise de decisão multicritério, recorreu-se ao método de análise hierárquica criado pelo professor Thoma L.Saaty.

O método AHP deste projeto foi elaborado de acordo com as sete fases definidas por Saaty.
Em primeiro lugar, definiu-se o problema e estruturou-se o diagrama hierárquico. O problema real estudado é a escolha do kit de desenvolvimento deste projeto. Os critérios estabelecidos para avaliar as hipóteses foram a Dificuldade de aprendizagem, a experiência da equipa de desenvolvimento e os recursos disponíveis para auxiliar no desenvolvimento da aplicação. Após alguma pesquisa e através dos critérios definidos, delinearam-se as quatro tecnologias possíveis, Tensorflow, CNTK, Caffe e Theano [4]. Na secção de Comparação de alternativas (ver secção 4.3) são apresentadas as tecnologias.

A Figura 2 apresenta o diagrama hierárquico para o problema estudado.

[1] *New product development*
[2] *Technology stage gate*

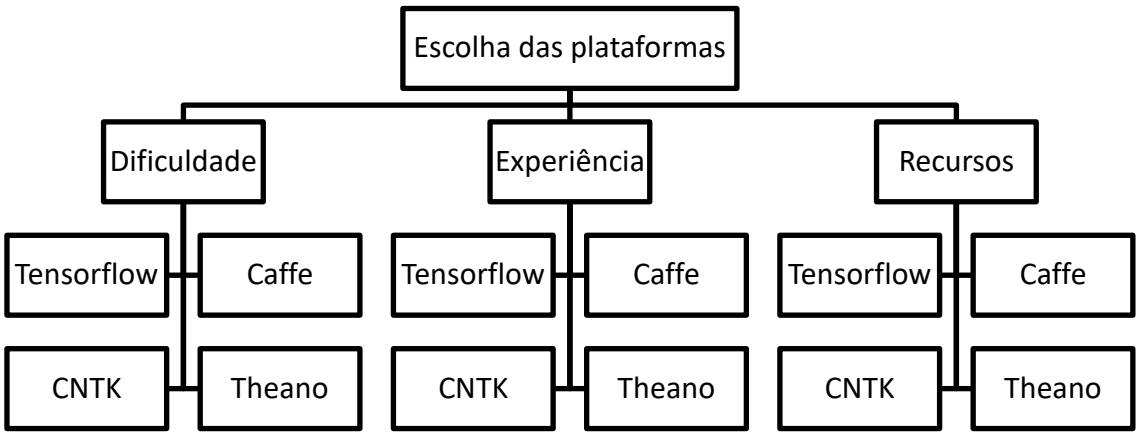


Figura 2 - Diagrama hierárquico das plataformas

Passou-se para a segunda fase, a fase da comparação das alternativas e critérios. Com o auxílio da escala fundamental definida por Satty (ver Tabela 1), foi possível estabelecer prioridades entre os elementos para cada nível da hierarquia (ver Tabela 2).

Tabela 1 - Escala fundamental de Satty

Nível de importância	**Definição**	**Explicação**
1	Igual importância	As duas atividades contribuem igualmente para o objetivo
3	Fraca importância	A experiência e o julgamento favorecem levemente uma atividade em relação à outra
5	Forte importância	A experiência e o julgamento favorecem fortemente uma atividade em relação à outra
7	Muito forte importância	Uma atividade é muito fortemente favorecida em relação a outra
9	Importância absoluta	A evidência favorece uma atividade em relação a outra com o mais alto grau de certeza
2, 4, 6, 8	Valores Intermediários	Quando se procura uma condição de compromisso entre duas definições

Tabela 2 - Matriz de Comparação dos critérios do Segundo Nível

	Dificuldade	**Experiência**	**Recursos**
Dificuldade	1/1	1/4	1/2
Experiência	4/1	1/1	2/1
Recursos	2/1	1/2	1/1

Através da matriz de comparação dos critérios (ver Tabela 2) é possível perceber, por exemplo, que a experiência nas tecnologias é um fator mais importante do que a Dificuldade.

Tabela 3 – Matriz de Comparação dos critérios do Segundo Nível com soma das colunas

	Dificuldade	**Experiência**	**Recursos**
Dificuldade	1/1	1/4	1/2
Experiência	4/1	1/1	2/1
Recursos	2/1	1/2	1/1
Soma	**7**	**7/4**	**7/2**

Tabela 4 – Matriz Normalizada dos Critérios do Segundo Nível

	Dificuldade	**Experiência**	**Recursos**
Dificuldade	1/7	4/28	2/14
Experiência	4/7	4/7	4/7
Recursos	2/7	4/14	2/7

Nesta fase, é necessário obter o vetor de prioridades que tem como principal objetivo a identificação da ordem de importância de cada critério. Para tal, procedeu-se ao calculo da média aritmética dos valores de cada linha da matriz normalizada (ver Tabela 5).

Tabela 5 – Vetor de Prioridades da Matriz Normalizada

	Dificuldade	**Experiência**	**Recursos**	**Prioridade Relativa**
Dificuldade	1/7	1/7	1/7	0.1429
Experiência	4/7	4/7	4/7	0.5714
Recursos	2/7	2/7	2/7	0.2857

Através dos resultados obtidos na Tabela 5 o critério "Experiência" surge em primeiro lugar, seguido do critério "Recursos" e "Dificuldade".

Na fase quatro, pretendeu-se avaliar a consistência das prioridades relativas. É então necessário calcular a Razão de Consistência (RC) de forma a avaliar se os julgamentos foram consistentes. Segundo Saaty, se o RC for menor que 10% então os julgamentos são coerentes.

Considera-se [Ax = λ_{max} x], no qual x é o vetor próprio.

$$\text{Ax} = \lambda max \text{ x} = \begin{vmatrix} 1 & 1/4 & 1/2 \\ 4 & 1 & 2 \\ 2 & 1/2 & 1 \end{vmatrix} \begin{vmatrix} 0.14 \\ 0.57 \\ 0.29 \end{vmatrix} \cong \lambda max \begin{vmatrix} 0.14 \\ 0.57 \\ 0.29 \end{vmatrix} =$$

$$= \begin{vmatrix} 0.43 \\ 1.71 \\ 0.86 \end{vmatrix} \cong \lambda max \begin{vmatrix} 0.14 \\ 0.57 \\ 0.29 \end{vmatrix}$$

Assim sendo, o valor próprio é calculado da seguinte forma:

$$\lambda \max = average\left\{\frac{0.43}{0.14} + \frac{1.71}{0.57} + \frac{0.86}{0.29}\right\} = 3.012$$

Após o cálculo de λmax, calcula-se o Índice de Consistência (IC) para que seja possível proceder ao cálculo da Razão de Consistência (RC), em que *n* é o número de critérios é o número de critérios.

$$\text{IC} = \frac{\lambda max - n}{n-1} = \frac{3.01-3}{3-1} = 0.005$$

A razão é calculada através da seguinte fórmula: $RC = \frac{IC}{IR}$

O valor de IR é valor obtido de acordo com a Figura 3.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

Figura 3 - Valores de IR para matrizes quadradas de ordem n

Uma vez que já temos calculado o valor de IC e uma vez que a nossa matriz é constituída por três critérios (IR = 0.58), é então possível calcular o valor de RC.

$$RC = \frac{IC}{0.58} = \frac{0.005}{0.58} = 0.009$$

Uma vez que o valor RC é menor que 0.1 (10%), pode-se concluir que os valores das prioridades relativas do exemplo utilizado estão consistentes.

Na fase cinco foi necessário fazer a construção da matriz de comparação paritária para cada critério em relação às alternativas selecionadas.

Tabela 6 – Matriz de Comparação do critério Dificuldade

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	3/1	2/1	5/1
Caffe	1/3	1	2/1	2/1
CNTK	1/2	1/2	1	3/1
Theano	1/5	1/2	1/3	1

Tabela 7 - Matriz de Comparação do critério Dificuldade com soma das colunas

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	3/1	2/1	5/1
Caffe	1/3	1	2/1	2/1
CNTK	1/2	1/2	1	3/1
Theano	1/5	1/2	1/3	1
Soma	**61/30**	**10/2**	**16/3**	**11**

Tabela 8 - Matriz Normalizada do critério Dificuldade

	Tensorflow	**Caffe**	**CNTK**	**Theano**	**Prioridade Relativa**
Tensorflow	30/61	3/5	6/16	5/11	**0.48**
Caffe	10/61	1/5	6/16	2/11	**0.23**
CNTK	15/61	1/10	3/16	3/11	**0.20**
Theano	6/61	1/10	1/16	1/11	**0.09**

Seguindo os mesmos passos para o cálculo da Matriz Normalizada do critério Dificuldade, foi possível calcular a Matriz Normalizada do critério Experiência (ver Tabela 11). Elaborou-se a matriz de comparação do critério Experiência (ver Tabela 9) e os respetivos cálculos para chegar à matriz normalizada (ver Tabela 10).

Tabela 9 – Matriz de Comparação do critério Experiência

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	5/1	1/2	5/1
Caffe	1/5	1	1/7	1/2
CNTK	2/1	7/1	1	7/1
Theano	1/5	2/1	1/7	1

Tabela 10 - Matriz de Comparação do critério Experiência com soma das colunas

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	5	2	5
Caffe	1/5	1	1/7	1/2
CNTK	2	7	1	7
Theano	1/5	2	1/7	1
Soma	**17/5**	**15**	**23/7**	**27/2**

Tabela 11 - Matriz Normalizada do critério Experiência

	Tensorflow	**Caffe**	**CNTK**	**Theano**	**Prioridade Relativa**
Tensorflow	5/17	1/3	14/23	10/27	**0.40**
Caffe	1/17	1/15	1/23	1/27	**0.05**
CNTK	10/17	7/15	7/23	14/27	**0.47**
Theano	1/17	2/15	1/23	2/27	**0.08**

Uma vez mais recorreu-se aos passos anteriormente descritos para obtermos a matriz normalizada do critério Recursos (ver Tabela 14). Nas tabelas seguintes são apresentados os passos necessários para a obtenção da matriz normalizada (ver Tabela 12 e Tabela 13).

Tabela 12 - Matriz de Comparação do critério Recursos

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	3/1	1/3	3/1
Caffe	1/3	1	1/4	1/2
CNTK	3/1	4/1	1	4/1
Theano	1/3	2/1	1/4	1

Tabela 13 - Matriz de Comparação do critério Recursos com soma das colunas

	Tensorflow	**Caffe**	**CNTK**	**Theano**
Tensorflow	1	3	1/3	3
Caffe	1/3	1	1/4	1/2
CNTK	3	4	1	4
Theano	1/3	2	1/4	1
Soma	**14/3**	**10**	**11/6**	**17/2**

Tabela 14 - Matriz Normalizada do critério Ferramentas

	Tensorflow	**Caffe**	**CNTK**	**Theano**	**Prioridade Relativa**
Tensorflow	3/14	3/10	6/33	6/17	**0.26**
Caffe	1/14	1/10	6/44	1/17	**0.09**
CNTK	9/14	2/5	6/11	8/17	**0.52**
Theano	1/14	1/5	6/44	2/17	**0.13**

Através do diagrama demonstrado em seguida é possível observar a importância relativa de cada uma das alternativas que compõe a estrutura hierárquica do problema em questão (ver Figura 4).

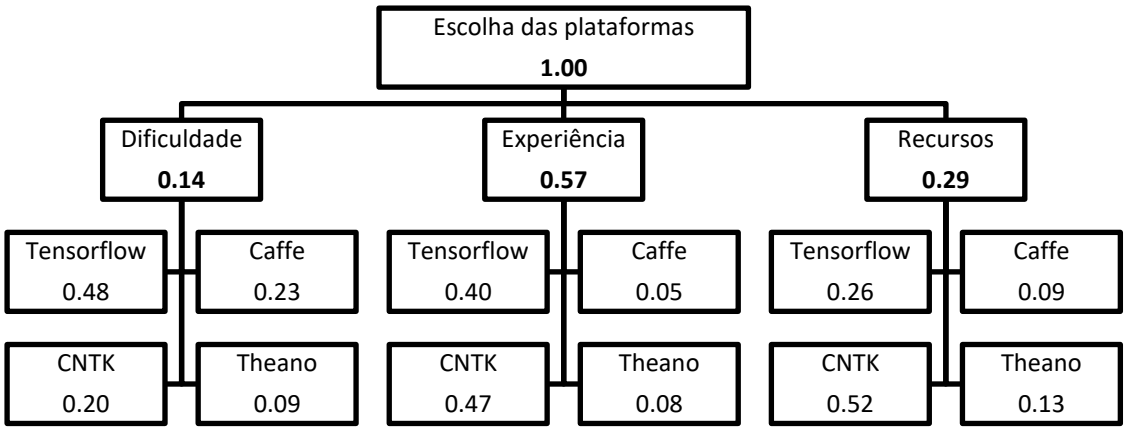


Figura 4 - Diagrama hierárquico com importâncias relativas associadas

Na etapa seis, obtemos as prioridades compostas das alternativas multiplicando os valores anteriores e os das prioridades relativos, obtidos no início do método. As linhas da matriz correspondem às alternativas e as colunas aos critérios adotados. Tensorflow, Caffe, CNTK e Theano é a ordem das alternativas e Dificuldade, Experiência e Recursos a ordem dos critérios [4].

$$\begin{vmatrix} 0.48 & 0.40 & 0.26 \\ 0.23 & 0.05 & 0.09 \\ 0.20 & 0.47 & 0.52 \\ 0.09 & 0.08 & 0.13 \end{vmatrix} * \begin{vmatrix} 0.14 \\ 0.57 \\ 0.29 \end{vmatrix} = \begin{vmatrix} 0.37 \\ 0.09 \\ 0.45 \\ 0.10 \end{vmatrix}$$

Por último, na fase sete escolhe-se a opção mais adequada de acordo com os critérios definidos inicialmente e as suas respetivas importâncias. Na Tabela 15 é possível perceber que a melhor opção é o CNTK.

Tabela 15 – Propriedades Compostas do problema

	Prioridades Compostas
Tensorflow	0.37
Caffe	0.09
CNTK	0.45
Theano	0.10

2.2.3 Perspetiva de valor

A análise de valor é um estudo organizado que se foca em perceber de que forma o produto pode ser melhorado ao mais baixo custo, sem sacrificar a fiabilidade e a qualidade do mesmo.

Segundo Woodruff [5], existem dois aspetos para caracterizar o valor para o cliente: valor desejado e valor percebido.

O valor de um produto poderá ter diferentes interpretações, dependendo do cliente. As suas características comuns são, o elevado nível de performance, potencial, impacto emocional, estética, entre outros, relativo ao seu custo. Pode ser caracterizado desta forma:

$$Valor = \frac{Performance + Potencial}{Custo} = \frac{Funcionalidade}{Custo}$$

O valor percebido é o benefício que o cliente acredita que recebeu de um produto após a sua compra. Isto é, o valor percebido é a opinião que o cliente formula, tendo em vista a satisfação das necessidades e exigências.

A baixo nível, o valor para o cliente depende das características do produto e do significado que essas características possuem. Num nível mais alto, o valor para o cliente pode ser visto como uma recompensa ou realização.

Baseado nos conceitos acima referidos, prevê-se que a comunidade surda vá beneficiar de um sistema que lhe permite traduzir praticamente todo o tipo de informação de uma forma acessível e rápida. A longo prazo poder-se-á desenvolver mais funcionalidades e incrementá-las a este sistema, facilitando assim a rotina diária da comunidade.

Numa perspetiva de futuro, desde que não existam alterações tecnológicas significativas e profundas, a comunidade poderá contar com um sistema estável, fiável e funcional.

2.2.4 Modelo *Canvas*

O quadro de modelo de negócio[3] é um modelo de gestão estratégica para o desenvolvimento de novos modelos de negócio. Este modelo permite tornar as propostas de valor do negócio visíveis e tangíveis, e fáceis de discussão e controlo. Integrar a proposta de valor num modelo de negócio viável é uma forma de adquirir valor para a organização.

Integrando perfeitamente o *Business Model Canvas* com a proposta de valor, as empresas focar-se-ão nos detalhes de como criar valor para os clientes. O modelo *Canvas* está representado num gráfico visual com elementos que descrevem a proposta de valor de uma empresa ou produto.

No total, o modelo *Canvas* é composto por nove elementos:

- Segmentos de clientes ou *Customer Segments*
- São grupos de clientes e/ou organizações que uma organização pretende servir e criar valor, através de uma proposta de valor orientada aos mesmos.
- Propostas de valor ou *Value Propositions*
- São baseadas numa panóplia de produtos ou serviços que criam valor para um determinado segmento de clientes.
- Canais ou *Channels*
- Descrevem como uma proposta de valor é comunicada e entregue a um segmento de mercado, através da comunicação, distribuição e canais de venda.
- Relacionamento com o Cliente ou *Customer Relationships*
- Determina que tipo de relação será estabelecida e mantida com cada segmento de clientes e nos quais é explicado como se obtêm novos clientes e se mantêm.
- Fluxos de rendimento ou *Revenue Streams*
- Resulta do sucesso da proposta de valor oferecida a um segmento de cliente. É a forma como a organização obtém valor com um preço que os clientes estão dispostos a pagar.
- Recursos-chave ou *Key Resources*
- São os recursos necessários para oferecer e entregar os elementos anteriormente descritos.
- Atividades-chave ou *Key Activities*

[3] *Business Model Canvas*

- São as atividades mais relevantes que a organização precisará de realizar com sucesso.

- Parceiros-chave ou *Key Partnerships*

- Refere-se à rede de fornecedores e parceiros que trarão recursos e atividades externas.

- Estrutura de custos ou *Cost Structure*

- Menciona todos os custos necessários para o funcionamento do modelo de negócio.

De seguida será apresentado o *Business Model Canvas* do modelo de negócio correspondente ao sistema em desenvolvimento. (ver Figura 5)

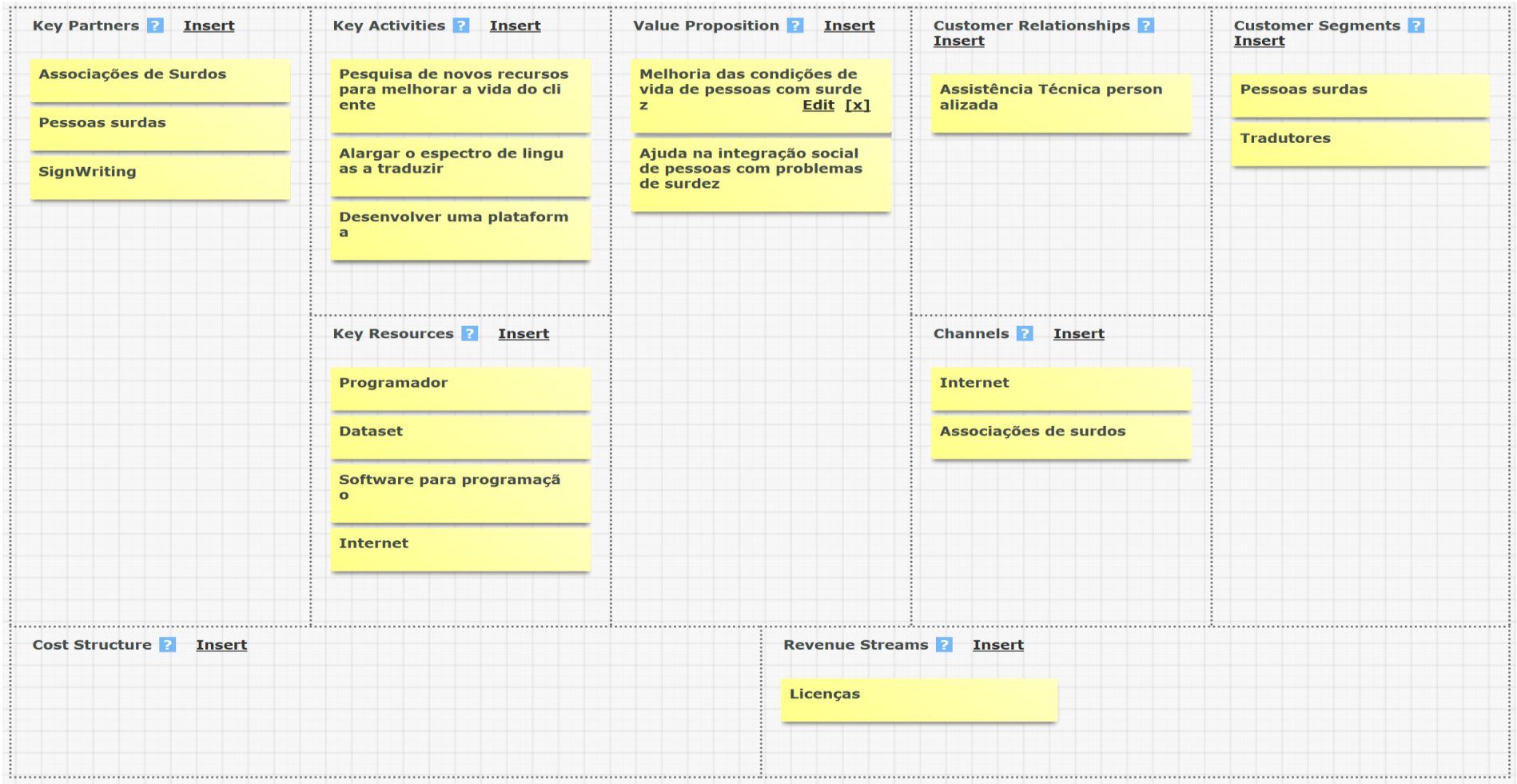


Figura 5 - *Business Model Canvas*

2.3 Estado da arte

O primeiro passo foi a necessidade de fazer uma pesquisa das tecnologias existentes e como estas dão resposta a este público alvo. Este estudo permitiu que o aluno tomasse conhecimento do que existe no mercado. Este conhecimento permitiu, igualmente, o surgimento de novas ideias. Nas próximas páginas, o aluno apresenta os resultados dessa pesquisa.

2.3.1 SignAll

SignAll [6] foi criado por János Rovnyai e Zsolt Robotka em Budapeste, Hungria, no ano 2012 [7]. Foi o conceito pioneiro de tradução automático de língua de sinais, baseando-se em técnicas como *computer vision* e *natural language processing* (*NLP*) para permitir a comunicação entre pessoas com problemas de audição ou surdos totais e pessoas sem problemas de audição.

O protótipo deste projeto permite traduzir parte de ASL para inglês.

2.3.2 SignWriting

SignWriting [8] é um sistema de escrita que usa símbolos para representar os movimentos das mãos e expressões faciais de línguas gestuais. É um alfabeto criado com o propósito de permitir melhores condições a pessoas com necessidades especiais.

Estes símbolos do alfabeto *SignWriting* são universais (ver Figura 6). Isto é, podem ser usados para escrever língua gestual Americana, língua gestual Portuguesa, língua gestual Francesa, entre outras.

Isto permite, finalmente, pessoas com surdez lerem um livro, as noticias no jornal, dicionários. Pode, igualmente, ser usado como ferramenta de ensino para principiantes de língua gestual ou pode ser usada para ensinar matemática ou história [9].

2.3.3 Google DeepMind

DeepMind [10] é atualmente líder mundial na inteligência artificial e emprega dezenas de equipas de investigadores que estão a tentar ultrapassar as limitações atuais da área, criando sistemas capazes de aprender a resolver qualquer tipo de problema.

Como exemplo mais recente desta determinação, foi demonstrado um sistema inteligente - AlphaGo [11] – capaz de jogar Go [12]. Este jogo é um dos mais complexos e intuitivos que existe atualmente e que contém mais jogadas possíveis do que átomos no universo [13]. AlphaGo foi testado contra o campeão mundial em titulo e venceu. Além deste sistema, *DeepMind* está a desenvolver um sistema capaz de “imaginar” as consequências das ações mesmo antes de as aplicar [14].

2.3.4 Microsoft Cognitive Toolkit

Microsoft Cognitive Toolkit (CNTK) [15] é um conjunto de ferramentas de *deep learning* (ver secção 2.4.1) que permite trabalhar com redes neuronais. CNTK permite que facilmente se implemente e combine diversos tipos de modelos, como por exemplo feed-forward DNNs, convolutional nets (CNNs), e recurrent neural network (RNNs/LSTMs). Implementa stochastic gradient descent (SGD, error backpropagation), aprendizagem com diferenciação automática e paralelização entre múltiplas GPUs e servidores.

2.4 Estado da arte em tecnologia relevante

2.4.1 Deep Learning

O deep learning [16] permite modelos computacionais que incluem múltiplas camadas de processamento para aprender representações de dados com múltiplos níveis de abstração. Esses métodos melhoraram drasticamente o estado da arte no reconhecimento de fala, reconhecimento e deteção de objetos e muitos outros domínios, como descoberta de medicamentos.

Esta técnica ajuda a descobrir mais facilmente a estrutura intrincada em grandes conjuntos de dados ao usar backpropagation para “ensinar” ao algoritmo como deve atualizar os seus parâmetros internos na sua execução.

Nos últimos anos, Deep learning tem ganho consistência. Esta consistência tem sido vista ao serem cada vez mais as áreas onde esta técnica é aplicada com sucesso.

2.4.2 Machine Learning

Machine learning (ML) é a técnica de inteligência artificial que faz com que os computadores executem o seu papel de forma natural sem que pareça que foram programados para tal. Se limitarmos a observação à última década, a aprendizagem automática foi responsável pelo surgimento de:

- carros automáticos
- recursos de reconhecimento de fala
- otimização das pesquisas na internet

Atualmente, a disseminação da ML é de tal forma grande que provavelmente qualquer pessoa a utiliza em diversos momentos do dia sem que haja uma perceção de tal interação. [17]

Os exemplos referidos anteriormente estão entre o estado da arte ligado a esta técnica. Mas, para além desses, é usado para:

- Categorizar imagens, principalmente de satélite e da área médica.
- Encontrar e assinalar possíveis fraudes bancárias
- Personalizar a recomendação de produtos, tanto em website como através de cupões ou promoções em loja, com base no comportamento do cliente
- Prever, com um elevado grau de precisão o estado meteorológico de uma dada região
- Tradução automática de texto e voz

2.4.3 Natural Language Processing

O processamento de língua natural (NLP) [18] teve inicio na década de 1950 com a interseção da inteligência artificial e da linguística. Na sua origem, o NLP era considerado um campo de estudos distinto da recuperação de informação de textos (IR).

No entanto, com o passar dos anos e com a evolução das técnicas, o NLP e a IR começaram a convergir e a usar metodologias idênticas e diversos pontos dos seus campos de estudo. [19]

3 Avaliar soluções e abordagens

3.1 Tecnologias

3.1.1 SignWriting

SignWriting é um standard internacional para língua gestual escrita, tanto de forma manual como de forma digital. Tem vindo a ser usada por uma larga comunidade internacional para os mais variados propósitos.

Entre esses propósitos:

- Educação
- Entretenimento
- Religião

Os textos traduzidos ao longo das últimas décadas [20] por este website serão uma peça essencial para conseguir obter os resultados esperados. O processo de limpeza e estruturação dos ficheiros obtidos é detalhado na secção 4.2.1.

A Figura 6 apresenta uma pequena parte do alfabeto Signwriting [21].

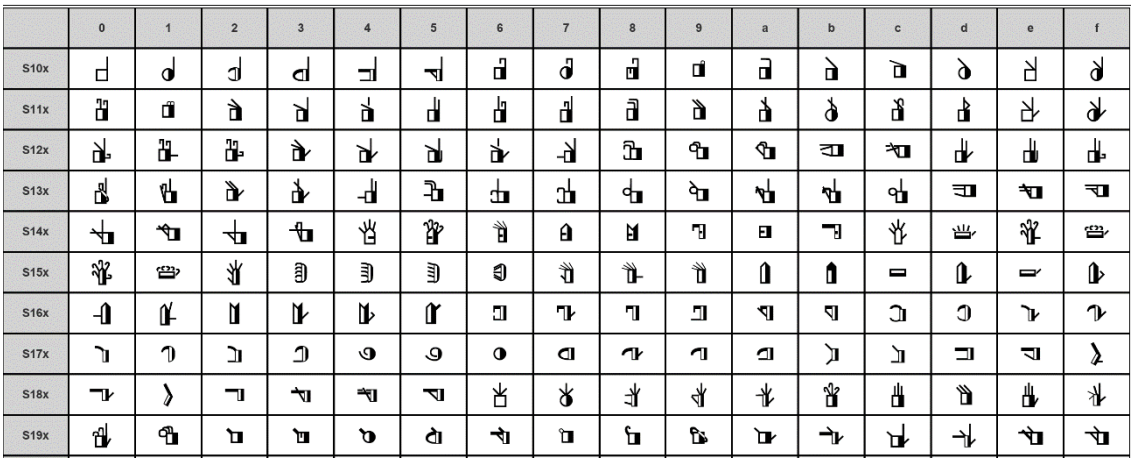


Figura 6 - Exemplo de caracteres do alfabeto Signwriting

3.1.2 CNTK

Esta é a ferramenta usada para, em conjunto com a linguagem de programação *Python,* implementar o código necessário para a criação da rede neuronal. A secção 2.3.4 apresenta uma breve descrição da plataforma. A secção 4.2 apresenta, de uma forma mais detalhada, todo o processo de implementação.

3.1.3 Algoritmo Sequence to Sequence

Cho et al. [22] introduz este algoritmo, que consiste em duas redes neuronais recorrentes (RNNs):

- Encoder, que vai fazer todo o processamento do input
- Decoder, que vai gerar o output

O input é introduzido num vetor de tamanho dinâmico, sendo este o único parâmetro enviado para o decoder. Bahdanau et al. [23] introduz uma alteração a este algoritmo, com a criação de um attention model.

A Figura 7 apresenta, de uma forma simplificada a arquitetura do algoritmo apresentado por Bahdanau et al.. O atention model, de forma simples, é usado para ajudar ambas as RNN (encoder e decoder) a alinhar o texto do input e do output.

É necessário obter um alinhamento entre input e output pois é com base neste alinhamento que o algoritmo vai fazer parte da sua aprendizagem. O attention model ajuda a criar este alinhamento, pois vai melhorando o alinhamento a cada passagem do algoritmo. [24]

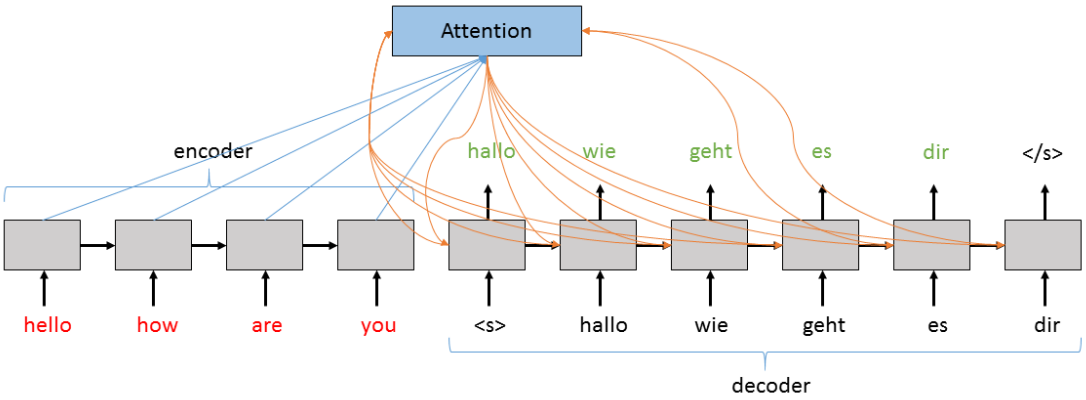


Figura 7 - Algoritmo Sequence to Sequence com attention model

A Figura 8 representa um algoritmo Sequence to Sequence com *Long-Short Term Memory* (*LSTM*) e com *attention model*.

Na secção 4.2 será apresentado, em maior detalhe, o *Attention model* e o *LSTM*.

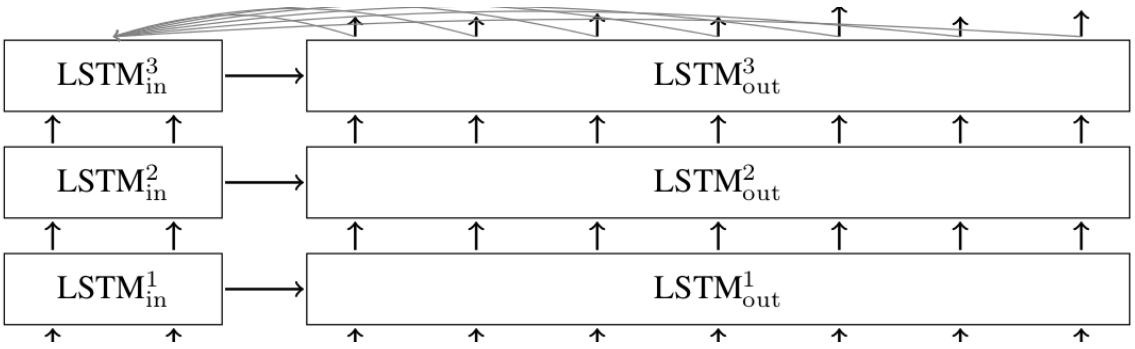


Figura 8 - Exemplo RNN com LSTM

3.1.4 Python

Python [25] é uma poderosa linguagem de programação orientada aos objetos, comparável com Perl, Ruby, Scheme ou Java.

Algumas características da linguagem:

- Fácil de implementar. Isso torna o Python ideal para desenvolvimento de protótipos e outras tarefas de programação, sem comprometer a capacidade de manutenção.

- Vem com uma grande biblioteca que suporta muitas tarefas comuns de programação, como conectar a servidores web, procura de texto com expressões regulares, leitura e modificação de ficheiros, entre muitas outras.

- É facilmente ampliado pela adição de novos módulos implementados numa linguagem compilada, como C ou C ++.

- Pode ser incorporado numa aplicação para fornecer uma interface programável.

- É executável nos principais sistemas operativos: Mac OS X, Windows, Linux e Unix.

3.1.5 VirtualSign

Trata-se de um tradutor em tempo real e bidirecional, que aproveita mais-valias da engenharia para cruzar potencialidades da inovação tenológica com a escrita e a Língua Gestual Portuguesa. Para os investigadores do GILT, esta proposta pode «revolucionar a comunicação com pessoas surdas em escolas, museus e outros locais, colocando a tecnologia ao serviço da comunidade». [26]

O VirtualSign [27] recorre a uma luva com sensores e à tecnologia Kinect, mas incorpora também um jogo didático para ensinar linguagem gestual. Três anos depois da sua aprovação, o VirtualSign está agora em fase de implementação. [28]

Figura 9 - Exemplo de animação do VirtualSign

3.1.6 Algoritmos de estimativa de qualidade

A estimativa de qualidade (EQ) tem como objetivo medir a qualidade do output da tradução automática (MT) sem traduções de referência. Geralmente, a EQ é abordada com vários recursos que indicam a influência, a adequação e a complexidade do par input-output. Tais recursos são usados em conjunto com métodos de machine learning. [29]

A QE é de crescente importância no campo da MT, uma vez que os sistemas MT são amplamente utilizados e a qualidade de cada frase traduzida pode variar consideravelmente. [30]

3.2 Abordagem

Esta secção tem como objetivo documentar de forma sucinta a abordagem ao problema, relatando as diferentes fases da construção deste projeto, de forma cronológica, para que seja possível compreender por que etapas e com que sequência este projeto se realiza desde o início até ao seu término. Este ponto reflete também o tipo de documentação de apoio que foi necessário ao seu desenvolvimento e em que circunstâncias foi aplicado.

A primeira fase deste projeto realiza-se com o primeiro dissecar do problema com o Professor Orientador. Nesta fase são abordados quais os resultados pretendidos com este projeto e qual a finalidade do mesmo.

Após a análise realizada no ponto anterior, é possível uma definição mais concreta dos objetivos, isto é, numa definição exata do tipo de análises a serem realizadas.

A segunda fase deste projeto consiste em fazer a pesquisa do estado da arte relativo à tradução para língua gestual. Durante este processo foram encontradas várias pessoas e equipas de interesse. De seguida, foi iniciado o contacto com os mesmos, através de email. O Anexo 5 – Lista de contactos efetuados tem a lista dos emails enviados de forma a iniciar contacto. Alguns dos emails estavam desatualizados e de outros nunca foi possível obter uma resposta. Após o inicio do contacto e caso obtivesse resposta, era então mantida correspondência de forma a receber algum tipo de informação ou dados relativos ao tema. Foi com um destes contactos que foi possível descobrir e ter acesso aos corpora usados na implementação deste projeto.

Após ter uma ideia aprofundada do estado da arte especifico da tradução para língua gestual, passou-se então para o estudo do estado da arte relacionado com as técnicas a implementar para resolver o problema em questão. Foi então feita uma pesquisa em artigos científicos, livros, publicações de revistas e websites sobre machine translation.

A quarta fase deste projeto consiste na aprendizagem da linguagem de programação Python. Após uma análise de algumas alternativas existentes e face a estas vantagens apresentadas, Python foi a linguagem escolhida para este projeto. Sendo Python uma matéria quase totalmente desconhecida, tornou-se imperativo um estudo intensivo sobre esta linguagem. A documentação de apoio neste processo de aprendizagem de Python: documentação oficial da linguagem; tutorias; papers; cursos online das conhecidas esco-

las Edx [31] e Coursera [32]. Ao mesmo tempo, foi encetado o estudo de deep learning, usando o mesmo tipo de estudo escolhido para aprender Python.

Na quinta fase é escolhida a língua gestual escrita a usar no desenvolvimento do projeto. Mesmo que o planeamento inicial e que o objetivo final sejam o uso de Língua gestual brasileira (LIBRAS) como principal língua do tradutor, foi escolhida ASL como língua do corpus. ASL é a língua gestual Americana e canadiana. A escolha recaiu sobre ASL devido ao maior volume de dados que a compõe. O maior volume de dados vai facilitar a aprendizagem, ajudando a perceber se o algoritmo está bem estruturado ou se tem alguma falha.

Com todas as condições reunidas que possibilitam o início do desenvolvimento deste projeto, inicia-se uma sexta fase que consiste na estruturação deste. Nesta fase é concebida uma solução capaz de responder aos requisitos necessários, assim como capaz de responder aos objetivos estipulados. Nesta fase é desenhada uma solução que permite obter uma maior segurança nos resultados obtidos.

Numa sétima fase, inicia-se a implementação da solução. Aqui é construída toda a solução idealizada na fase anterior, desde a recolha, tratamento e limpeza dos corpura à criação, treino e teste do modelo de tradução.

Na oitava e última fase, após concluída toda a solução, encontram-se reunidas as condições ideais para uma análise geral de resultados. Nesta fase será feita uma apreciação global de resultados obtidos com os modelos treinados.

4 Design

4.1 Design da solução

Tendo em conta que o sistema a implementar não terá qualquer interação com o utilizador comum, a necessidade da criação de qualquer tipo de interface com o utilizador é mínima. Isto é, o sistema proposto tem como objetivo a criação de um modelo de tradução e fazer o seu treino, de forma a reduzir ao máximo a taxa de erro. Assim que essa taxa de erro se encontre abaixo de um valor considerado aceitável, é então integrado com VirtualSign.

Assim sendo, esta aplicação encontra-se dividida em duas camadas:

- Gestão de dados: responsável por todo o processo de limpeza e transformação dos dados que alimentam a rede neuronal

- Inteligência: contêm todas as implementações necessárias para que sejam aplicados os algoritmos construídos que irão produzir os resultados pretendidos, sobre os dados presentes nos corpora.

4.2 Arquitetura

Como pode ser visto na Figura 10, esta ilustra a metodologia do sistema e as suas partes constituintes. Esta figura permite compreender também como as diversas componentes deste projeto se relacionam.

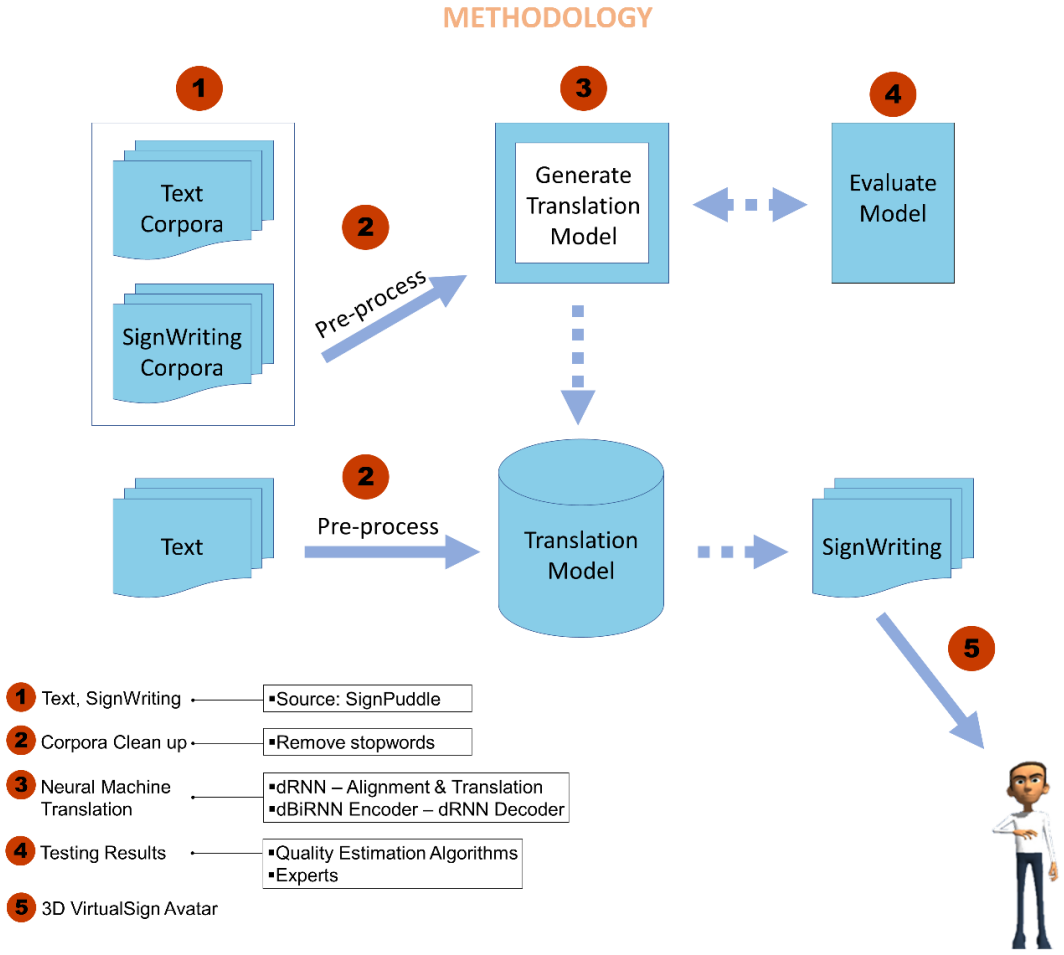


Figura 10 - Metodologia

4.2.1 Pré-processamento

A Figura 11 demonstra o estado dos corpora que foram obtidos. Incluem campos que não tem qualquer valor para o sistema (`<src></src>` e `<vídeo></vídeo>`). Assim sendo, o primeiro passo é criar e aplicar um algoritmo de limpeza, de forma a obter o resultado que a Figura 12 demonstra.

Após a remoção de tags sem valor para o projeto, é feita uma avaliação do estado dos corpora. É então notado que, em alguns casos, o texto em lingua natural e a tradução para signwriting estão armazenados em tags com o mesmo nome. Ou existe mais que uma tradução e mais que um texto dentro da mesma tag `<entry>`.

Noutros casos, existe uma tradução, mas não existe um texto. Por vezes existem erros ortográficos nos textos. Por vezes, os textos são apenas um marcador, sem qualquer referência à sua origem.

Após identificar todos estes casos, foi possível criar algoritmos específicos que os corrigem. De forma a corrigir erros ortográficos, foi usada a biblioteca autocorrect (`from autocorrect import spell`) que permite descobrir os erros ortográficos e oferece uma sugestão de correção.

Foram encontrados centenas de marcadores nos corpora. Estes marcadores foram, mais tarde identificados como passagens da bíblia (em especifico de duas versões: NLT - New Living Translation e KJB – King James Bible). Foi então criado um algoritmo suportado por python que identificava o capitulo e o versículo e pesquisava o texto correspondente na internet [33].

Após todo o processo de pré-processamento, obtém-se um ficheiro estruturado como o exemplo da Figura 12.

O passo seguinte é processar estes dados e criar o ficheiro *.ctf. De forma a gerar os corpora neste formato, é usado um script python disponibilizado pelo CNTK [34].

4.2.1.1 Fonte de dados

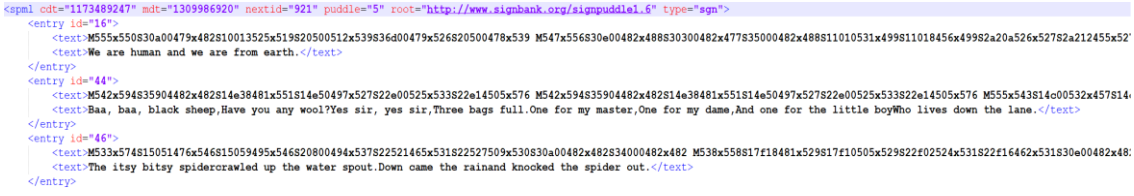
Corpora estruturado em ficheiros xml [35] obtidos no site signwriting.com.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE spml SYSTEM "http://www.signpuddle.net/spml_1.6.dtd">
<spml root="http://www.signbank.org/signpuddle1.6" type="sgn" puddle="5" cdt="1173489247" mdt="1309986920" nextid="921">

  <text>M521x542830a00482x482834400482x482810011500x512820e00485x521822b01462x498 L552x568830c00482x465835f00482x46581ce17525x51681ce1f448x515826c02521x541826c1a460x541836d10479x502 S387
  <term><![CDATA[literature (n)]]></term>
  <png>iVBORw0KGgoAAAANSUhEUgAAACQAAAA8CAMAAAA5W+hcAAAABGdBTUEAANbY1E9YMgAAABlORVh0U29mdHdhcmUAQWRvYmUgSW1hZ2VSZWFkeXHJZTwAAAAVUExURf///wAAAP8AAAAzMzM/wAz///MzGCTyE4AAADASURBVHja7JPBDsM
<entry id="1" next="2" cdt="1173489247" mdt="1232231641" usr="">
  <term>M611x54381f720367x501820320390x50181dc20407x48682f900423x492810001481x507826500592x479810000596x48982ff00452x48382ed00465x489826a06566x516830004512x482810e02532x52082f900593x471<
  <text>M538x553830a00482x48281f720467x538820320493x53881dc20511x52382f900527x530 M530x564818049472x528818041505x528820500487x553820500504x553830c00482x482 M536x552830c00482x482832d00482
  <term><![CDATA[ASL Facial Adverbs 1]]></term>
  <src><![CDATA[Written by Dr. Karen Van Hoek. Signed in ASL by George Butch Zein.]]></src>
</entry>

<entry id="2" top="1" next="3" prev="1" cdt="1173489385" mdt="1232231656" usr="">
  <term>M596x54381f720377x503820320400x50381dc20417x48682f900431x49482ff00457x48382ed00468x488810001480x510830004507x482810e02522x520826a06555x516810e00581x487826500582x46682f900584x458<
  <text>M518x579830a00482x482810004499x530822a04503x55582f700506x572 M538x569830a00482x482819a00510x524819a08466x524822a04517x550822a14475x55082fb04496x563 M533x517830c00482x482815a10521
  <term><![CDATA[ASL Facial Adverbs 2]]></term>
  <src><![CDATA[Written by Dr. Karen Van Hoek. Signed in George Butch Zein.]]></src>
</entry>
```


Figura 11 - Exemplo demonstrativo da estrutura dos corpora obtidos



```
<spml cdt="1173489247" mdt="1309986920" nextid="921" puddle="5" root="http://www.signbank.org/signpuddle1.6" type="sgn">
    <entry id="16">
        <text>M555x550S30a00479x482S10013525x519S20500512x539S36d00479x526S20500478x539 M547x556S30e00482x488S30300482x477S35000482x488S11010531x499S11018456x499S2a20a526x527S2a212455x52
        <text>We are human and we are from earth.</text>
    </entry>
    <entry id="44">
        <text>M542x594S35904482x482S14e38481x551S14e50497x527S22e00525x533S22e14505x576 M542x594S35904482x482S14e38481x551S14e50497x527S22e00525x533S22e14505x576 M555x543S14c00532x457S14
        <text>Baa, baa, black sheep,Have you any wool?Yes sir, yes sir,Three bags full.One for my master,One for my dame,And one for the little boyWho lives down the lane.</text>
    </entry>
    <entry id="46">
        <text>M533x574S15051476x546S15059495x546820800494x537822521465x531822527509x530830a00482x482S34000482x482 M538x558S17f18481x529S17f10505x529822f02524x531822f16462x531S30e00482x48
        <text>The itsy bitsy spidercrawled up the water spout.Down came the rainand knocked the spider out.</text>
    </entry>
```


Figura 12 - Exemplo dos corpora após a fase de limpeza

4.2.1.2 Formato de dados

Os corpora resultantes do pré-processamento estão já no formato CNTKTextFormatReader [36]. De seguida é possivel ver um exemplo onde a sequencia de input (S0) se encontra na coluna da esquerda e a sequencia de output (S1) na direita:

```
0 |S0 3:1 |# <s>       |S1 3:1 |# <s>
0 |S0 23315:1 |# Disneyland  |S1 11243:1 |#
M526x556S2ff00482x483S10110507x490S2e200506x524
0 |S0 1:1 |# </s>      |S1 1:1 |# </s>
```

4.2.2 Criação do modelo de tradução

O trabalho do modelo é aprender o mapeamento da sequência de entrada para a sequência de saída que ele irá gerar. O trabalho do encoder é apresentar uma boa representação da entrada que o decoder pode usar para gerar uma boa saída. Tanto para o encoder como para o decoder, o LSTM traz um bom desempenho.

Ao configurar sequências, existem dois eixos dinâmicos que são importantes a considerar. O primeiro é o eixo do batch, que é o eixo ao longo do qual várias sequências são agrupadas. O segundo é o eixo dinâmico especifico para essa sequência.

Os eixos dinâmicos são particularmente importantes no mundo das RNN. Em vez de ter que decidir um comprimento de sequência máximo antes do tempo, levando a um desperdício de recursos, os eixos dinâmicos da CNTK permitem que os comprimentos de sequência variáveis sejam tão eficientes quanto possível.

Por exemplo, sequence to sequence, temos duas sequências: a sequência de entrada e a sequência de saída. Uma das coisas que torna esse tipo de rede tão poderoso é que o comprimento da sequência de entrada e a sequência de saída não precisam se corresponder. Portanto, tanto a sequência de entrada como a sequência de saída requerem seu próprio eixo dinâmico exclusivo.

A rede Sequence to Sequence é um encoder RNN (LSTM) seguido de um decoder RNN (LSTM) e uma camada de saída densa. O método create_model() (ver Anexo 6 – Código) cria as funções CNTK de encoder e decoder. A função de decoder usa diretamente a função de encoder e o valor de retorno de create_model() é a função CNTK decoder.

Na Figura 13 vemos um diagrama de como funciona a versão em camadas da rede Sequence to Sequence com attention model. Como é possível reparar, a saída de cada camada do codificador e decodificador é usada como entrada para a camada seguinte. O Attention model concentra-se na camada superior do encoder e informa a primeira camada do decoder.

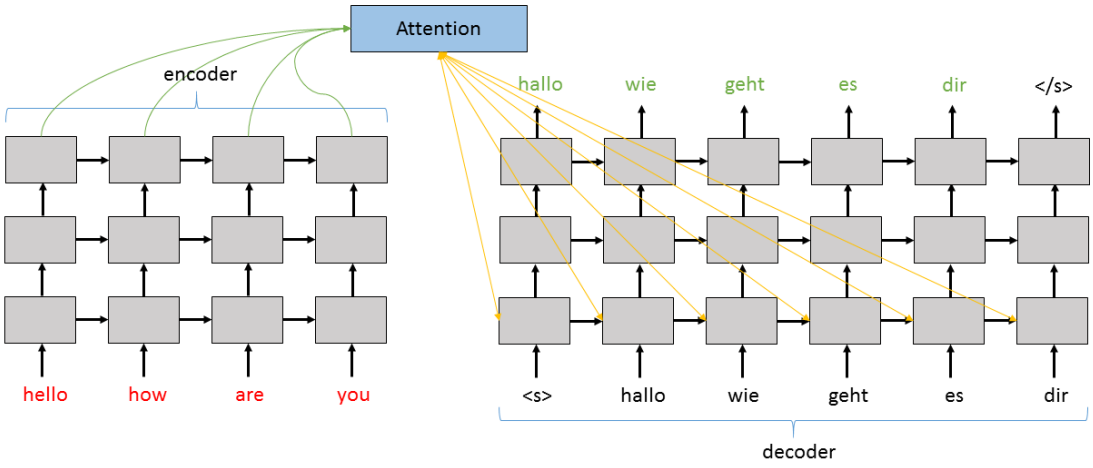


Figura 13 - Exemplo de funcionamento do algoritmo Sequence to Sequence com Attention model

O modelo criado pode ser visto como um modelo "abstrato". Neste caso, vai ser usado primeiro para criar uma versão de treino do modelo, e de seguida vai ser usado para criar uma versão onde o *decoder* será a saída do *Hardmax* da rede.

É agora criado um método `create_criterion_function`, que vai ser usado para calcular a função de perda `cross_entropy_with_softmax` e obter o `classification_error`, que retorna a percentagem de erro por palavra.

Anteriormente foi criada uma versão do modelo de treino e uma versão do modelo para avaliação. Normalmente, esta última versão não seria necessária neste momento, mas irá ser usada para que possa mostrar periodicamente como o modelo está a evoluir.

Em seguida, serão configuradas algumas variáveis padrão necessárias para o ciclo de treino. É definida a taxa de aprendizagem inicial, um `learner` usando o algoritmo `adam_sgd` e um `learning_rate_schedule,` que reduz lentamente a taxa de aprendizagem. E por fim, é criado o objeto `Trainer`.

Dentro do ciclo de treino, é solicitado o próximo minibatch. No entanto, a cada 30 minibatch é executada uma avaliação usando o model_greedy da rede. É executada uma única sequência, "grammar", de forma a verificar o que o modelo está a prever.

4.2.3 Avaliar o modelo

De forma a avaliar a evolução do modelo durante o treino, numa primeira fase, é usada a taxa de erro como forma de avaliação. O CNTK oferece o método `ProgressPrinter` que calcula os valores médios e absolutos de taxa de erro e de perda.

Numa fase mais avançada do projeto, é necessário implementar os algoritmos de estimativa de erro. Estes ajudarão a obter uma maior precisão no treino do modelo, assim como uma maior precisão da qualidade da tradução obtida com o modelo, a cada momento.

4.2.4 Produzir resultados

Após passar todas as fases anteriormente descritas, é então altura de obter resultados. Como forma mais simples de obter dados, é produzido um output em *SignWriting*.

Visto que o sistema apenas aprendeu a sua forma codificada (`M526x556S2ff00482x483S10110507x490S2e200506x524`), é necessário converter este output para os símbolos correspondentes em língua gestual. Para esse efeito é usado uma pequena libraria em *javascript*, disponibilizada pelos criadores do alfabeto *SignWriting*. [37]

Como é possível verificar na Figura 14, o output gerado é convertido para um sinal ou conjunto de sinais em SignWriting.

Figura 14 - Exemplo da conversão do output para símbolo

4.3 Comparação de alternativas

A Figura 15 demonstra a comparação em termos de velocidade de treino em diferentes condições, entre as seguintes plataformas e a escolhida para implementação deste projeto (CNTK).

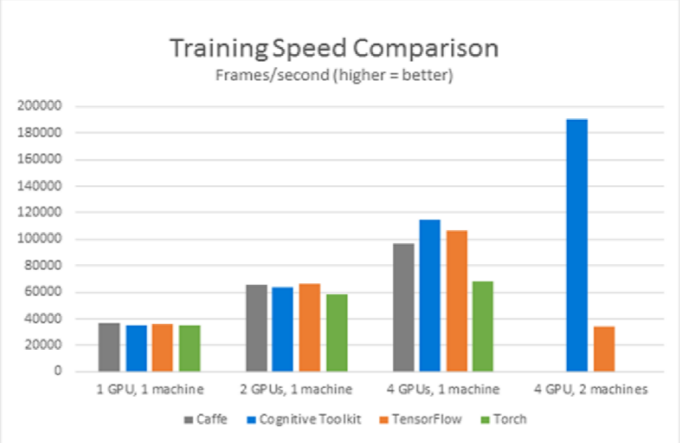


Figura 15 - Gráfico comparativo da velocidade de treino entre plataformas

4.3.1 Tensorflow

TensorFlow [38] é uma biblioteca para computação numérica. A arquitetura flexível permite que o utilizador paralelize a execução do seu código para um ou mais CPUs ou GPUs, localizados num servidor, computador pessoal ou mesmo um dispositivo móvel com uma única API.

Foi originalmente desenvolvido por investigadores e engenheiros que trabalham no departamento de Machine Intelligence da Google, com o objetivo de realizar pesquisas com maior rapidez e precisão, mas o sistema é genérico o suficiente para ser aplicável em uma grande variedade de domínios.

Exemplos de empresas que usam o Tensorflow:

- Ebay
- Google
- Intel
- Snapchat
- Twitter
- Uber

4.3.2 Theano

Theano [39] é uma biblioteca Python que permite definir, otimizar e avaliar expressões matemáticas envolvendo matrizes multidimensionais de forma eficiente.

Características de Theano:

- Integração com NumPy
- Uso facilitado da GPU para processamento
- Otimização de velocidade e estabilidade
- MILA vai parar de desenvolver Theano e a próxima versão será a última versão [40].

4.3.3 Caffe

Caffe: Convolutional Architecture for Fast Feature Embedding [41] é uma estrutura de deep learning desenvolvida a pensar em velocidade e modularidade. É desenvolvido pelo

Berkeley AI Research (BAIR) [42] e pela comunidade. Yangqing Jia criou o projeto durante o seu doutoramento na UC Berkeley.

5 Avaliação

5.1 Descrição das experiências e avaliação a realizar à solução preconizada

5.1.1 Que grandezas vai utilizar para avaliar o seu trabalho?

5.1.1.1 Tempo

Tem de ser um sistema que seja de resposta rápida. O utilizador dependerá da resposta deste sistema para poder executar alguma tarefa/responder a alguma pergunta.

5.1.1.2 Exatidão

Tem de ser um sistema com um grau de exatidão o mais próximo de 100% possível. Isto deve-se ao facto de ser um sistema de tradução. Uma tradução mal efetuada ou com um grau de exatidão menor, irá provocar problemas ao utilizador.

5.1.1.3 Satisfação do utilizador

Serão desenvolvidos inquéritos ao utilizador, de forma a apurar o grau de satisfação do mesmo. É importante apurar o quão satisfeito está o utilizador. Caso esteja insatisfeito, significa que o sistema está a funcionar de forma errática e deverá ser apurado o motivo.

5.1.2 Que hipótese pretende testar para suportar os resultados do seu trabalho?

Permitir que o sistema criado produza traduções de diversas frases não incluídas nos corpora.

5.1.3 Qual a metodologia de avaliação?

5.1.3.1 Grupos de controlo/teste

Visto que este sistema está a ser focado para responder a um problema especifico de um grupo restrito de pessoas, é importante envolve-los no design, desenvolvimento e avaliação. Deste modo, é imperativo que as associações locais e nacionais se envolvam, de forma a divulgar o sistema e a abranger um maior numero de pessoas.

5.1.3.2 Resultados inquérito de satisfação

Após cada ronda de testes, o grupo de controlo será sujeito a um inquérito de satisfação, de forma a puder ser feita uma análise mais aprofundada sobre o estado de cada versão.

5.2 Avaliação de Resultados

Após finalizar a implementação do código, é então a altura para treinar o algoritmo e obter resultados. Devido ao facto de ser extremamente demorado a executar cada epoch (cerca de 6 horas) e de necessitar de demasiados recursos computacionais (cerca de 96% de utilização do CPU a cada instante), tornou-se impossível obter mais resultados.

Mas, ainda assim, é possível realizar algumas análises sobre estes.

O modelo que foi treinado durante 1 epoch obteve os seguintes resultados:

- Perda – 5.41
- Taxa de erro - 50%

O modelo que foi treinado durante 2 epoch obteve os seguintes resultados:

- Perda – 5.26
- Taxa de erro – 49%

A taxa de erro de ambos os modelos deixa transparecer que o modelo treinado até ao momento está aquém da espectativa. Isto é, a taxa de erro é demasiado alta para um qualquer sistema de machine translation implementado e que se pretenda disponibilizar para uso diário da população.

Sendo que este tipo de tradução depende ainda mais da precisão do que a tradução entre duas línguas naturais, é então necessário executar um maior número de epoch para que a taxa de erro continue a baixar.

A implementação de um algoritmo de estimativa de qualidade (ver 3.1.6) é algo que foi planeado numa primeira fase, mas devido à já grande complexidade do algoritmo atual, ficou reservado para uma segunda fase do projeto. Este tipo de algoritmo é ideal para ajudar a reduzir a enorme taxa de erro encontrada com apenas duas epoch de treino.

Visto que o modelo criado ainda não tem uma taxa de erro aceitável, não foi realizada nenhuma das avaliações previamente descritas neste capitulo.

6 Conclusões e trabalho futuro

6.1 Contribuições

Durante o tempo de desenvolvimento deste projeto, foram surgindo algumas oportunidades de aprendizagem e apresentação de resultados.

EPIA é uma conferencia internacional de inteligência artificial [43]. Foi possível, ao aluno, fazer a apresentação de um poster com a metodologia que este propõe para atacar o problema (ver Anexo 2). De forma a ser possível esta apresentação, foi necessário entregar um breve artigo cientifico (ver Anexo 1 – Paper de admissão EPIA) com uma breve descrição do projeto e da proposta de resolução.

Após a aceitação do mesmo, o aluno teve a possibilidade de afixar o poster e interagir com diversas pessoas de renome mundial na área da inteligência artificial (ver Anexo 3 - Presença na EPIA).

Por último, o aluno teve a possibilidade de ver o ser poster afixado na Noite Europeia do Investigador, sessão que se realizou na cidade do porto (ver Anexo 4 – Poster NEI).

6.2 Principais conclusões desta dissertação

Este estudo permitiu conhecer melhor as tecnologias de deep learning e machine translation, assim como uma visão sobre a língua gestual e a cada vez maior necessidade de se tornar acessível à população em geral.

Após a implementação e a execução, mesmo que limitada do projeto, é possível ver que este poderá ser o elo de ligação que falta para tornar a inclusão social da pessoa surda uma realidade. É possível reparar que o deep learning trouxe uma melhoria da precisão dos modelos criados com machine learning em geral, e com machine translation em particular.

Tendo isto em conta, a capacidade computacional mundial está, neste momento, nivelada com a capacidade dos algoritmos de aprenderem e descobrirem mais padrões, que ajudarão o ser humano a evoluir na sua vida.

6.3 Trabalho futuro

Entende-se por trabalhos futuros, encontrar meios para solucionar os problemas encontrados na implementação do protótipo.

É necessário executar mais epoch no modelo de treino, de forma a melhorar a taxa de erro e verificar se a implementação atual está de acordo com o pretendido.

É igualmente necessário implementar a conexão entre o output do algoritmo e o Virtual-Sign.

A implementação dos algoritmos de estimativa de qualidade deve ser incluída nesta lista, pois são necessários à melhoria da taxa de erro.

Após terminar o treino com ASL, deve ser expandido para LIBRAS.

7 Referências

[1] “Associação de Surdos da Alta Extremadura,” [Online]. Available: http://www.associacaosurdosaltaestremadura.org/index.php?id=44. [Acedido em 11 2016].

[2] Sutton, “Signwriting,” [Online]. Available: http://www.signwriting.org/symposium/presentation0040.html . [Acedido em 08 2017].

[3] “europarl,” [Online]. Available: http://www.statmt.org/europarl/. [Acedido em 11 2016].

[4] S. Nicola, “Análise de Valor,” [Online]. Available: https://moodle.isep.ipp.pt/pluginfile.php/143572/mod_resource/content/1/An%C3%A1lise_Valor_Aula4.pdf. [Acedido em 15 Fevereiro 2017].

[5] “Woodruff, R.B. J. of the Acad. Mark. Sci. (1997) 25: 139. doi:10.1007/BF02894350,” [Online]. Available: http://link.springer.com/article/10.1007%2FBF02894350. [Acedido em Janeiro 2017].

[6] “SignAll,” [Online]. Available: http://www.signall.us/. [Acedido em April 2017].

[7] “crunchbase,” [Online]. Available: https://www.crunchbase.com/organization/signall. [Acedido em September 2017].

[8] “SignWriting,” [Online]. Available: http://www.signwriting.org. [Acedido em December 2016].

[9] “Signwriting,” [Online]. Available: http://www.signwriting.org/about/what/what02.html. [Acedido em May 2017].

[10] “deepmind,” [Online]. Available: https://deepmind.com/. [Acedido em January 2017].

[11] “deepmind.com,” [Online]. Available: https://deepmind.com/research/alphago/. [Acedido em October 2017].

[12] “go-portugal.org,” [Online]. Available: http://www.go-portugal.org/go/o-que-e-o-go. [Acedido em October 2017].

[13] “senseis.xmp.net,” [Online]. Available: https://senseis.xmp.net/?NumberOfPossibleGoGames. [Acedido em October 2017].

[14] “wired.co.uk,” [Online]. Available: http://www.wired.co.uk/article/googles-deepmind-creates-an-ai-with-imagination. [Acedido em October 2017].

[15] “github.com/Microsoft/CNTK,” [Online]. Available: https://github.com/Microsoft/CNTK. [Acedido em December 2016].

[16] “nature,” [Online]. Available: http://www.nature.com/nature/journal/v521/n7553/abs/nature14539.html?foxtrotcallback=true. [Acedido em February 2017].

[17] “Coursera,” [Online]. Available: https://pt.coursera.org/learn/machine-learning. [Acedido em October 2017].

[18] “ncbi,” [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3168328/. [Acedido em August 2017].

[19] F. J. D. Nitin Indurkhya, Handbook of Natural Language Processing, Second Edition.

[20] “signwriting,” [Online]. Available: http://www.signwriting.org/symposium/presentation0040.html . [Acedido em September 2017].

[21] “slevinski,” [Online]. Available: https://slevinski.github.io/SuttonSignWriting/characters/symbols.html#?ui=en&set=code. [Acedido em November 2016].

[22] "arxiv:1406.1078," [Online]. Available: https://arxiv.org/abs/1406.1078.

[23] "arxiv:1409.0473," [Online]. Available: https://arxiv.org/abs/1409.0473. [Acedido em May 2017].

[24] R. C. Mikhail Mikhailov, Corpus Linguistics for Translation and Contrastive Studies: A guide for research, Routledge, 2016.

[25] "wiki.python," [Online]. Available: https://wiki.python.org/moin/BeginnersGuide/Overview. [Acedido em August 2017].

[26] "isep," [Online]. Available: http://www.isep.ipp.pt/new/viewnew/4146. [Acedido em October 2017].

[27] "VirtualSign," [Online]. Available: http://193.136.60.223/virtualsign/pt/index.php. [Acedido em October 2017].

[28] "fct," [Online]. Available: https://www.fct.pt/apoios/projectos/consulta/vglobal_projecto?idProjecto=121878&idElemConcurso=4231. [Acedido em October 2017].

[29] V. L. G. H. P. F. B. D. B. F. B. L. S. Kashif Shah, "SHEF-NN: Translation Quality Estimation with Neural Networks".

[30] H. Kim e J.-H. Lee, "A Recurrent Neural Networks Approach for Estimating the Quality of Machine Translation Output".

[31] "edx," [Online]. Available: https://www.edx.org/.

[32] "coursera," [Online]. Available: https://pt.coursera.org/.

[33] "biblehub," [Online]. Available: http://biblehub.com/. [Acedido em August 2017].

[34] "github.com/Microsoft," [Online]. Available: https://github.com/Microsoft/CNTK/tree/master/Scripts. [Acedido em September 2017].

[35] "w3schools," [Online]. Available: https://www.w3schools.com/xml/ . [Acedido em October 2017].

[36] "docs.microsoft," [Online]. Available: https://docs.microsoft.com/en-us/cognitive-

] toolkit/brainscript-cntktextformat-reader.

[37 “slevinski,” [Online]. Available: https://slevinski.github.io/SuttonSignWriting/guide.html.
] [Acedido em December 2016].

[38 “tensorflow,” [Online]. Available: https://www.tensorflow.org/.
]

[39 “deeplearning,” [Online]. Available: http://deeplearning.net/software/theano/. [Acedido
] em October 2017].

[40 “groups.google,” [Online]. Available: https://groups.google.com/forum/#!msg/theano-
] users/7Poq8BZutbY/rNCIfvAEAwAJ. [Acedido em October 2017].

[41 “arxiv:1408.5093,” [Online]. Available: https://arxiv.org/abs/1408.5093. [Acedido em
] October 2017].

[42 “caffe.berkeleyvision,” [Online]. Available: http://caffe.berkeleyvision.org/. [Acedido em
] September 2017].

[43 “epia2017,” [Online]. Available: https://web.fe.up.pt/~epia2017/.
]

[44 “Associação de Surdos do Porto,” [Online]. Available: www.asurdosporto.org.pt.
]

[45 A. Osterwalder, Value Proposition Design: How to create products and services customers
] want, New Jersey: Wiley, 2014.

[46 L. F. P. D. Michael H. Morris, Business-to-Business Marketing: A Strategic Approach,
] Londres: SAGE, 2001.

[47 N. Solutions, “Value Analysis and Function Analysis System Technique,” [Online].
] Available: http://www.npd-solutions.com/va.html. [Acedido em 13 Fevereiro 2017].

[48 N. Kokemuller, “What Is Customer Perceived Value?,” [Online]. Available:
] http://smallbusiness.chron.com/customer-perceived-value-23692.html. [Acedido em 13
Fevereiro 2017].

[49 “Associação de Surdos da Alta Extremadura,” [Online]. Available:
] http://www.associacaosurdosaltaestremadura.org/. [Acedido em 11 2016].

8 Anexos

8.1 Anexo 1 – Paper de admissão EPIA

Translating Written Language to Sign Language using Parallel Corpora

Diogo Oliveira[1], Nuno Escudeiro[1,2], Paula Escudeiro[1,3]

[1] Instituto Superior de Engenharia do Porto, Porto, Portugal
[2] Laboratory of Artificial Intelligence and Decision Support, LIAAD/INESC TEC, Porto, Portugal
[3] Games, Interaction and Learning Technologies, GILT, Porto, Portugal

Abstract. Deaf people suffer from info and social exclusion due to difficulties in communicating with non-deaf people. Equity and inclusion will be promoted if there are solutions assisting a more effective communication between deaf and non-deaf communities. We expect to provide accurate translations from written to sign languages through a sustainable and scalable process to facilitate the communication between deaf and non-deaf people in daily life.

- **Keywords:** Deaf, Sign Writing, Translation, Parallel Corpora, Deep neural networks, Neural Machine Translation.

1 Interpreting the problem

1.1 Context

The Portuguese Sign Language (LGP) is officially recognized by law and it is the first language for 33.000 people [1]. Portuguese and LGP – like any other written and sign languages – are two unique languages, different from each other in all linguistic aspects. The communication between deaf and non-deaf people is very ineffective because both use distinct native languages. This scenario is common to all countries in the world creating a significant barrier to the social inclusion of the deaf community.

Translation to sign language requires experts in sign language translation and, frequently, video production. It is a demanding and costly process that requires a significant amount of specialized resources. This is an expensive solution and not feasible for daily life.

1.2 Problem

To promote equal opportunities and the inclusion of deaf people in active social life it is strictly mandatory to develop sustainable and scalable solutions for the translation from written language to sign language that might be available for daily life.

Sign Writing is a system of writing sign languages that can be used as a vehicle for the automatic translation between written and sign languages. Sign writing together

with automatic translation will assist the daily communication and the creation of digital content to the hearing-impaired at low cost and permanently available helping the social inclusion of the deaf community.

1.3 Aims

We aim to generate accurate digital content in sign language for the hearing-impaired people through a fast and cheap process that might be sustainable and scalable.

The ultimate goal is to provide equal opportunities and promote the social inclusion of the deaf or hearing-impaired people.

2 Methodology

The methodology we are implementing includes several stages from corpora acquisition to evaluation. The core stages are briefly described below.

2.1 Compile parallel corpora

We have compiled parallel corpora in written Brazilian Portuguese and LIBRAS sign writing – the Brazilian sign language – from SignPuddle [2]. We have decided to use Brazilian Portuguese due to the difficulty in finding Portuguese sign writing sources.

2.2 Generate translation model

The automatic translation will be based on Parallel Corpora techniques. We will use two aligned corpus, one with documents in plain text and the other with the counterparts in sign writing. We will learn the translation model using Deep Neural Networks and Neural Machine Translation. The final aim is to align by word but, in an early stage will be base our translation in sentence alignment techniques.

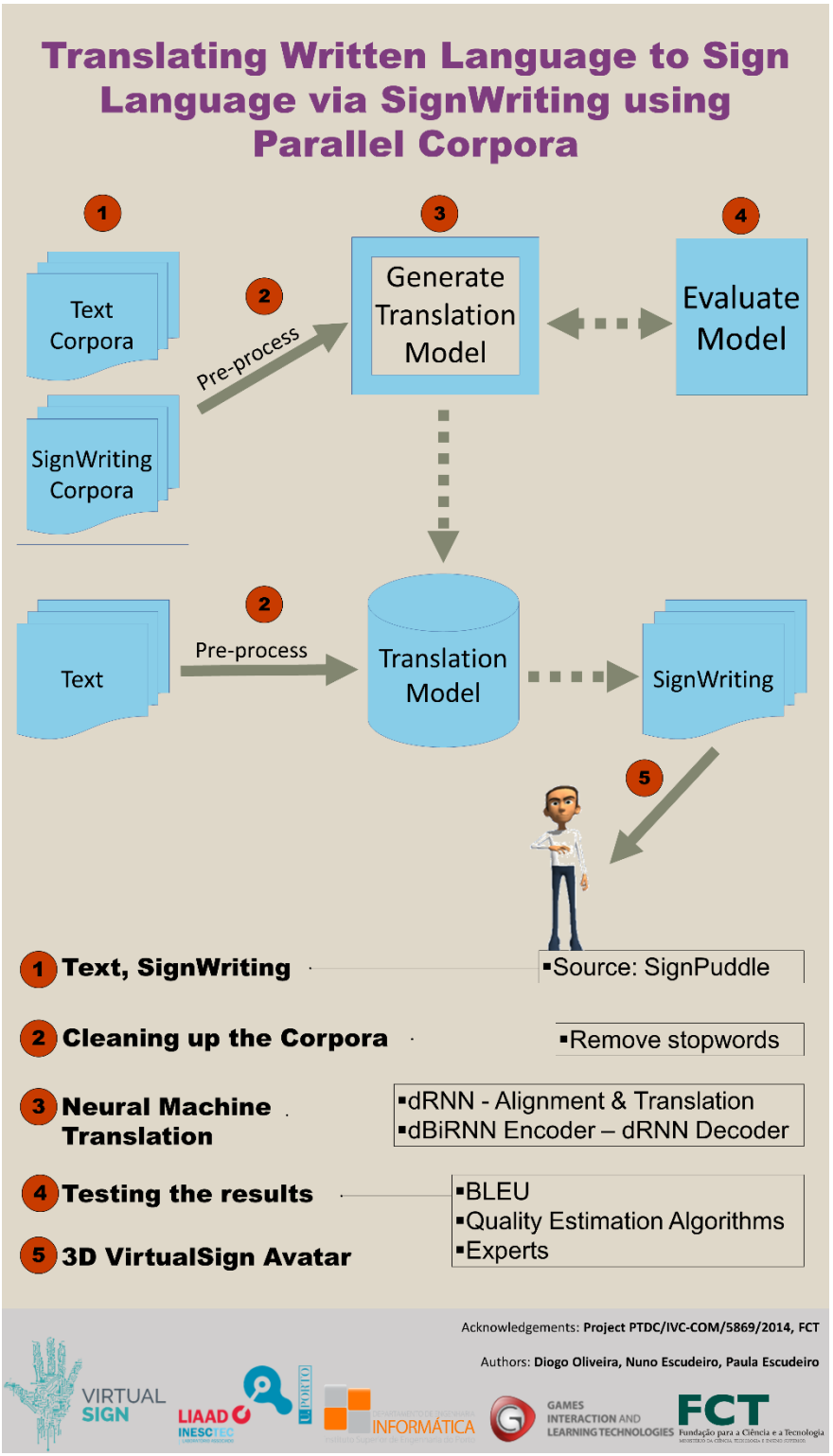
2.3 Output

The automatic translator will generate sign language translated documents in sign writing. Then we will use a 3D avatar to represent the corresponding gestures in sign language. The final output will be an animation where the avatar mimics the sign language translation corresponding to the written text. This output will be validated by a group of Brazilian deaf users and sign language translators.

References

1. Deaf Association of Porto Homepage, http://www.asurdosporto.org.pt/, last accessed 2017/04/19.
2. SignPuddle Homepage, http://www.signbank.org/signpuddle/, last accessed 2017/04/18.

8.2 Anexo 2 – Poster EPIA



8.3 Anexo 3 - Presença na EPIA

8.4 Anexo 4 – Poster NEI

DEAFNY - Translating Written Language to Sign Language via SignWriting

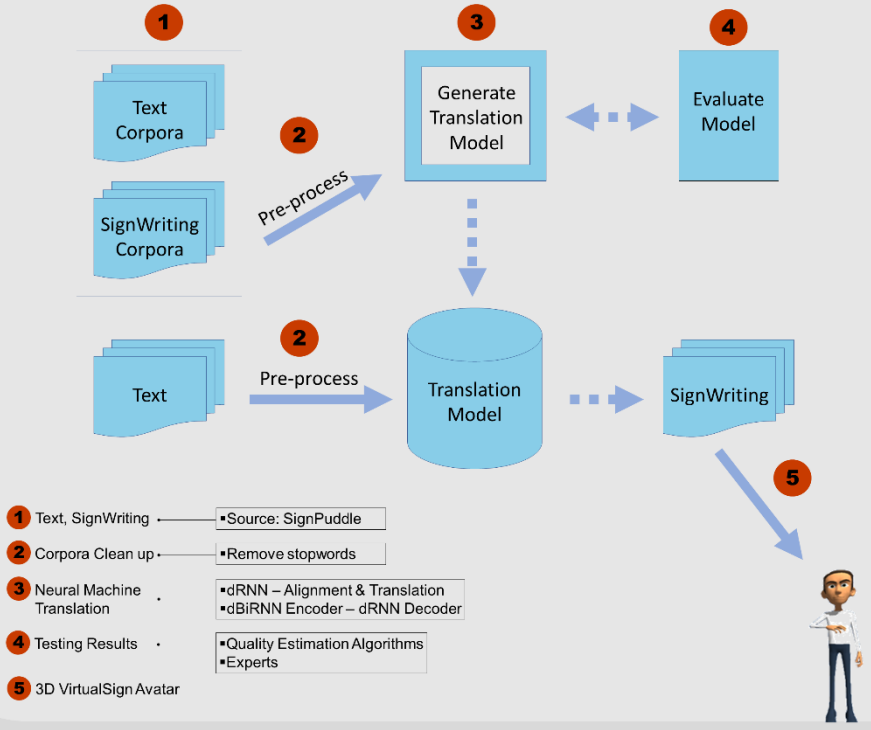
ABSTRACT

Hearing impaired people suffer from info and social exclusion due to difficulties in communicating with non-deaf people. Equity and inclusion will be promoted if there are solutions assisting a more effective communication between deaf and non-deaf communities.
We expect to provide accurate translations from natural to sign languages through a sustainable and scalable process to facilitate the communication between deaf and non-deaf people in daily life.

GOALS

We aim to generate accurate digital content in sign language for the hearing-impaired people through a fast and cheap process that might be sustainable and scalable.
The ultimate goal is to provide equal opportunities and promote the social inclusion of the deaf or hearing-impaired people.

METHODOLOGY



Acknowledgements: **Project PTDC/IVC-COM/5869/2014, FCT**
Authors: **Diogo Oliveira, Nuno Escudeiro, Paula Escudeiro**

VIRTUAL SIGN · LIAAD INESCTEC · U.PORTO · DEPARTAMENTO DE ENGENHARIA INFORMÁTICA · GAMES INTERACTION AND LEARNING TECHNOLOGIES · FCT Fundação para a Ciência e a Tecnologia

8.5 Anexo 5 – Lista de contactos efetuados

10/11/2016 - Valerie Sutton(Sutton@SignWriting.org) [respondeu]

10/11/2016 - http://www.eud.eu/contact-us/?tab=written-message

10/11/2016 - Dr. Reiner Konrad from Institute for German Sign Language and Communication of the Deaf(reiner.konrad@sign-lang.uni-hamburg.de) [respondeu]

10/11/2016 - http://deeplearning.net/ (admin@deeplearning.net) [não foi entregue, email não existe]

17/11/2016 - hello@signall.us

17/11/2016 - pkoehn@inf.ed.ac.uk

06/02/2017 - Microsoft Research (facebook)

07/02/2017 - asg-info@microsoft.com (envio sem subject do email)

23/02/2017 - asg-info@microsoft.com [recebida]

13/03/2017 - mtsl@letras.up.pt;

13/03/2017 - carol@bu.edu

14/03/2017 - nals@contato.ufsc.br

14/03/2017 - http://corpuslibras.ufsc.br/contato

14/03/2017 - las@fe.up.pt

18/04/2017 - jorg.tiedemann@lingfil.uu.se

8.6 Anexo 6 – Código

```
from __future__ import print_function
import os
from cntk import Trainer, Axis
from cntk.io import MinibatchSource, CTFDeserializer, StreamDef, StreamDefs,
INFINITELY_REPEAT
from cntk.learners import momentum_sgd, fsadagrad, momen-
tum_as_time_constant_schedule, learning_rate_schedule, UnitType
from cntk import input, cross_entropy_with_softmax, classification_error,
sequence, element_select, alias, hardmax, placeholder, combine, parameter,
times, plus
from cntk.ops.functions import CloneMethod, load_model, Function
from cntk.initializer import glorot_uniform
from cntk.logging import log_number_of_parameters, ProgressPrinter
from cntk.logging.graph import plot
from cntk.layers import *
from cntk.layers.sequence import *
from cntk.layers.models.attention import *
from cntk.layers.typing import *
import numpy as np
import time

hidden_dim = 512
num_layers = 6
attention_dim = 128
use_attention = True
use_embedding = True
embedding_dim = 200
max_epochs = 2
epoch_size = 908241
length_increase = 1.5
model_path = "model_20.cmf"
data_dir = "."

def get_vocab(path):
    vocab = ([w.strip() for w in open(vocab_file).readlines()])
    i2w = { i:w for i,w in enumerate(vocab) }

    return (vocab, i2w)

def create_reader(path, randomize, sparse=True, size=INFINITELY_REPEAT):
    return MinibatchSource(CTFDeserializer(path, StreamDefs(
        features  = StreamDef(field='S0', shape=input_vocab_dim,
is_sparse=sparse),
        labels    = StreamDef(field='S1', shape=label_vocab_dim,
is_sparse=sparse)
    )), randomize=randomize, max_samples = size)

def create_model():
    embed = C.layers.Embedding(embedding_dim, name='embed') if
use_embedding else identity

    with C.layers.default_options(enable_self_stabilization=True,
go_backwards=not use_attention):
        LastRecurrence = C.layers.Fold if not use_attention else
C.layers.Recurrence
        encode = C.layers.Sequential([
            embed,
```

```
            C.layers.Stabilizer(),
            C.layers.For(range(num_layers-1), lambda:
                C.layers.Recurrence(C.layers.LSTM(hidden_dim))),
            LastRecurrence(C.layers.LSTM(hidden_dim), re-
turn_full_state=True),
            (C.layers.Label('encoded_h'), C.layers.Label('encoded_c')),
        ])

    with C.layers.default_options(enable_self_stabilization=True):
        # sub-layers
        stab_in = C.layers.Stabilizer()
        rec_blocks = [C.layers.LSTM(hidden_dim) for i in range(num_layers)]
        stab_out = C.layers.Stabilizer()
        proj_out = C.layers.Dense(label_vocab_dim, name='out_proj')
        # attention model
        if use_attention: # maps a decoder hidden state and all the encoder
states into an augmented state
            attention_model = C.layers.AttentionModel(attention_dim,
name='attention_model') # :: (h_enc*, h_dec) -> (h_dec augmented)
        # layer function
        @C.Function
        def decode(history, input):
            encoded_input = encode(input)
            r = history
            r = embed(r)
            r = stab_in(r)
            for i in range(num_layers):
                rec_block = rec_blocks[i]   # LSTM(hidden_dim)  # :: (dh,
dc, x) -> (h, c)
                if use_attention:
                    if i == 0:
                        @C.Function
                        def lstm_with_attention(dh, dc, x):
                            h_att = atten-
tion_model(encoded_input.outputs[0], dh)
                            x = C.splice(x, h_att)
                            return rec_block(dh, dc, x)
                        r = C.layers.Recurrence(lstm_with_attention)(r)
                    else:
                        r = C.layers.Recurrence(rec_block)(r)
                else:
                    # unlike Recurrence(), the RecurrenceFrom() layer takes
the initial hidden state as a data input
                    r =
C.layers.RecurrenceFrom(rec_block)(*(encoded_input.outputs + (r,))) # :: h0,
c0, r -> h
            r = stab_out(r)
            r = proj_out(r)
            r = C.layers.Label('out_proj_out')(r)
            return r

    return decode

def create_model_train(s2smodel):
    # model used in training (history is known from labels)
    # note: the labels must not contain the initial <s>
    @Function
    def model_train(input, labels): # (input*, labels*) --> (word_logp*)
```

```
        # The input to the decoder always starts with the special label se-
quence start token.
        # Then, use the previous value of the label sequence (for training)
or the output (for execution).
        # BUGBUG: This will currently fail with sparse input.
        past_labels = Delay(initial_state=sentence_start)(labels)
        return s2smodel(past_labels, input)
    return model_train

def create_model_greedy(s2smodel):
    # model used in (greedy) decoding (history is decoder's own output)
    @Function
    @Signature(InputSequence[Tensor[input_vocab_dim]])
    def model_greedy(input): # (input*) --> (word_sequence*)

        # Decoding is an unfold() operation starting from sentence_start.
        # We must transform s2smodel (history*, input* -> word_logp*) into
a generator (history* -> output*)
        # which holds 'input' in its closure.
        unfold = UnfoldFrom(lambda history: s2smodel(history, input) >>
hardmax,
                            until_predicate=lambda w:
w[...,sentence_end_index],  # stop once sentence_end_index was max-scoring
output
                            length_increase=length_increase)
        return unfold(initial_state=sentence_start, dynamic_axes_like=input)
    return model_greedy

def create_criterion_function(model):
    @Function
    @Signature(input = InputSequence[Tensor[input_vocab_dim]], labels = La-
belSequence[Tensor[label_vocab_dim]])
    def criterion(input, labels):
        # criterion function must drop the <s> from the labels
        postprocessed_labels = sequence.slice(labels, 1, 0) # <s> A B C
</s> --> A B C </s>
        z = model(input, postprocessed_labels)
        ce   = cross_entropy_with_softmax(z, postprocessed_labels)
        errs = classification_error      (z, postprocessed_labels)
        return (ce, errs)

    # use the following to render the Function graph to a PDF file
    #plot(criterion, filename=os.path.join(MODEL_DIR, "model") + '.pdf')
    return criterion

def Evaluator(model, criterion):
    from cntk import Trainer
    from cntk.learners import momentum_sgd, learning_rate_schedule, Unit-
Type, momentum_as_time_constant_schedule
    loss, metric = Trainer._get_loss_metric(criterion)
    parameters = set(loss.parameters)
    if model:
        parameters |= set(model.parameters)
    if metric:
        parameters |= set(metric.parameters)
    dummy_learner = momentum_sgd(tuple(parameters),
                                 lr = learning_rate_schedule(1, Unit-
Type.minibatch),
```

```
                                    momentum = momen-
tum_as_time_constant_schedule(0))
    return Trainer(model, (loss, metric), dummy_learner)

def evaluate_metric(reader, s2smodel, num_minibatches=None):

    model_train = create_model_train(s2smodel)
    criterion = create_criterion_function(model_train)

    evaluator = Evaluator(None, criterion)

    # Get minibatches of sequences to test and perform testing
    minibatch_size = 1024
    total_samples = 0
    total_error = 0.0
    while True:
        mb = reader.next_minibatch(minibatch_size)
        if not mb: # finish when end of test set reached
            break
        #mb_error = evaluator.test_minibatch(mb[reader.streams.features],
mb[reader.streams.labels])
        mb_error = evaluator.test_minibatch({criterion.arguments[0]:
mb[reader.streams.features], criterion.arguments[1]:
mb[reader.streams.labels]})
        num_samples = mb[reader.streams.labels].num_samples

        total_error += mb_error * num_samples
        total_samples += num_samples

        rate = total_error/total_samples
        print("error rate of {:.1f}% in {} samples".format(100 * rate, to-
tal_samples))

        if num_minibatches != None:
            num_minibatches -= 1
            if num_minibatches == 0:
                break

    # and return the test error
    rate = total_error/total_samples
    print("error rate of {:.1f}% in {} samples".format(100 * rate, to-
tal_samples))
    return rate

def train(train_reader, valid_reader, vocab, i2w, s2smodel, max_epochs,
epoch_size):

    # Note: We would like to set the signature of 's2smodel'
(s2smodel.update_signature()), but that will cause
    # an error since the training criterion uses a reduced sequence axis
for the labels.
    # This is because it removes the initial <s> symbol. Hence, we must
leave the model
    # with unspecified input shapes and axes.

    # create the training wrapper for the s2smodel, as well as the criteri-
on function
    model_train = create_model_train(s2smodel)
    criterion = create_criterion_function(model_train)
```

```
    # also wire in a greedy decoder so that we can properly log progress on
a validation example
    # This is not used for the actual training process.
    model_greedy = create_model_greedy(s2smodel)

    # This does not need to be done in training generally though
    # Instantiate the trainer object to drive the model training

    #tensorboard_writer = TensorBoardProgressWriter(freq=30, log_dir='log',
model=s2smodel)

    minibatch_size = 72
    lr = 0.001 if use_attention else 0.005   # TODO: can we use the same
value for both?
    learner = fsadagrad(model_train.parameters,
                        lr       = learn-
ing_rate_schedule([lr]*2+[lr/2]*3+[lr/4], UnitType.sample, epoch_size),
                        momentum = momentum_as_time_constant_schedule(1100),
                        gradient_clipping_threshold_per_sample=2.3,
                        gradient_clipping_with_truncation=True)
    trainer = Trainer(None, criterion, learner)
    #trainer = Trainer(None, criterion, learner, tensorboard_writer)

    # Get minibatches of sequences to train with and perform model training
    total_samples = 0
    mbs = 0
    eval_freq = 100

    # print out some useful training information
    log_number_of_parameters(model_train) ; print()
    progress_printer = ProgressPrinter(freq=30, tag='Training')

    for epoch in range(max_epochs):
        while total_samples < (epoch+1) * epoch_size:
            # get next minibatch of training data
            mb_train = train_reader.next_minibatch(minibatch_size)
            #train-
er.train_minibatch(mb_train[train_reader.streams.features],
mb_train[train_reader.streams.labels])
            trainer.train_minibatch({criterion.arguments[0]:
mb_train[train_reader.streams.features], criterion.arguments[1]:
mb_train[train_reader.streams.labels]})

            progress_printer.update_with_trainer(trainer, with_metric=True)
# log progress

            # every N MBs evaluate on a test sequence to visually show how
we're doing
            if mbs % eval_freq == 0:
                mb_valid = valid_reader.next_minibatch(1)

                # run an eval on the decoder output model (i.e. don't use
the groundtruth)
                e = model_greedy(mb_valid[valid_reader.streams.features])

#print(format_sequences(sparse_to_dense(mb_valid[valid_reader.streams.featu
res]), i2w))
                print("->")
                print(format_sequences(e, i2w))
```

```
                # debugging attention
                if use_attention:
                    debug_attention(model_greedy,
mb_valid[valid_reader.streams.features])

            total_samples +=
mb_train[train_reader.streams.labels].num_samples
            mbs += 1

        # log a summary of the stats for the epoch
        progress_printer.epoch_summary(with_metric=True)
        s2smodel.save("model{0}.cmf".format(epoch))
        #evaluate_metric(test_reader, s2smodel)

    # done: save the final model
    s2smodel.save(model_path)
    print("%d epochs complete." % max_epochs)

def format_sequences(sequences, i2w):
    return [" ".join([i2w[np.argmax(w)] for w in s]) for s in sequences]

def debug_attention(model, input):
    q = C.combine([model, model.attention_model.attention_weights])
    words, p = q(input) # Python 3
    output_seq_len = words[0].shape[0]
    p_sq = np.squeeze(p[0][:output_seq_len,:,:]) # (batch, output_len, in-
put_len, 1)
    opts = np.get_printoptions()
    np.set_printoptions(precision=5)
    print(p_sq)
    np.set_printoptions(**opts)


start_time = time.time()

valid_file = os.path.join(data_dir, 'tiny.ctf')
train_file = os.path.join(data_dir, 'cmudict-0.7b.train-dev-20-21.ctf')
test_file = os.path.join(data_dir, 'cmudict-0.7b.test.ctf')
vocab_file = os.path.join(data_dir, 'cmudict-0.7b.mapping')

# Read vocabulary data and generate their corresponding indices
vocab, i2w = get_vocab(vocab_file)

# Train data reader
input_vocab_dim  = len(vocab)
label_vocab_dim  = len(vocab)

# hook up data
train_reader = create_reader(train_file, True)
valid_reader = create_reader(valid_file, False)
test_reader = create_reader(test_file, False)

sentence_start =  Constant(np.array([w=='<s>' for w in vocab],
dtype=np.float64))
sentence_end_index = vocab.index('</s>')

# Source and target inputs to the model
inputAxis = Axis('inputAxis')
labelAxis = Axis('labelAxis')
```

```
InputSequence = SequenceOver[inputAxis]
LabelSequence = SequenceOver[labelAxis]

model = create_model()
train(train_reader, valid_reader, vocab, i2w, model, max_epochs, epoch_size)

model = Function.load("model1.cmf")


#evaluate_metric(test_reader, model)

elapsed_time = time.time() - start_time

print("Elapsed time: {}".format(hms_string(elapsed_time)))
```