



Universidade do Minho
Escola de Engenharia

Inteligência Ambiente e Assistentes Virtuais Educação em Saúde

Ricardo Araújo

Abstract: O século XXI foi decididamente marcado pela inércia que a tecnologia impregnou na sociedade, mas também por uma perda crescente de paciência por parte dos indivíduos que a utilizam. Neste contexto, as horas de sono surgem como vítima fácil, pelas mais variadas razões, sendo contudo necessário sensibilizar e informar o público geral dos problemas bem estudados que esta prática pode causar.

Docentes: Prof. Paulo Novais, Prof. Dalila Alves Durães

Data: 17 de maio de 2024

1 Introdução

No contexto da UC de Ambientes Inteligentes foi desenvolvido um chatbot, recorrendo à ferramenta RASA [1], capaz de responder, de forma pertinente, a perguntas por parte do utilizador, relativas ao seu hábitos de sono. Este chatbot será ainda capaz de prestar informação sobre a temática do sono, de forma relativamente informal, e enunciar, caso solicitado, as suas fontes de informação.

2 Rasa

A versão Open Source da ferramenta RASA permite desenvolver assistentes com um grau limitado de inteligência, lidando com algumas problemáticas com relativa facilidade.

2.1 Domínio

De forma a cingir a capacidade de resposta do chatbot a desenvolver, surge a necessidade de definir um domínio de atuação. A ferramenta Rasa permite definir concretamente estes limites através do ficheiro 'domain.yml'. Posto isto foram postuladas as intenções do utilizador passíveis de ser identificadas, as possíveis respostas a estas intenções, e quais as ações a tomar no decorrer do diálogo.

2.2 Intenções do utilizador

Uma das problemáticas na estruturação de um diálogo prende-se obviamente com a interpretação, por parte do computador, de qual a intenção expressa pelo utilizador. Esta interpretação estará sujeita aos exemplos definidos para cada intenção definidos no documento 'nlu.yml' e ainda aos parâmetros definidos no documento 'config.yml'.

2.2.1 Definição de intenções

Neste trabalho foram definidas diversos exemplos para várias intenções (intents), de modo a tornar a interação mais fluída e dotar o chatbot de uma boa capacidade de resposta. Deste modo foram definidos as seguintes intenções:

<code>greet</code>	Interação de salutaçāo inicial.
<code>mood_great</code>	Percepção de um estado positivo do humor do utilizador.
<code>mood_unhappy</code>	Percepção de um estado negativo do humor do utilizador.
<code>sleep_question_1</code>	Questão 1 : Quantas horas deve dormir por noite.
<code>sleep_question_2</code>	Questão 2 : Quantos padrões diferentes de sono se passa durante uma noite de sono.
<code>sleep_question_3</code>	Questão 3 : Quais as consequências de não dormir o suficiente.
<code>sleep_question_4</code>	Questão 4 : Como melhorar o sono.
<code>sleep_facts</code>	Interpelar o chatbot a apresentar factos relevantes sobre o sono.
<code>affirm</code>	Interações de confirmação
<code>want_sources</code>	Interpelar o chatbot a apresentar as suas fontes de informação.
<code>not_eager_to_learn</code>	Percepção de desinteresse, por parte do utilizador, de questionar mais o chatbot.
<code>out-of-scope</code>	Identificar questões fora do domínio preestabelecido.
<code>need_help</code>	Interpelar o chatbot a apresentar as suas funcionalidades.
<code>thanking</code>	Percepção de um agradecimento por parte do utilizador.
<code>goodbye</code>	Intenção de despedida.
<code>bot_challenge</code>	Interpelar o chatbot a apresentar a sua natureza artificial.

2.2.2 Classificação de intenções

De modo a classificar as intenções previamente definidas é necessário definir 2 elementos.

O primeiro trata-se de um documento, "nlu.yml", que contem cada intenção e diferentes exemplos, de modo a possibilitar o treino do modelo.

O segundo diz respeito ao ficheiro de configuração, "config.yml", onde é necessário definir uma pipeline de processamento da query introduzida pelo utilizador, tendo isto sido definido da seguinte forma:

Como podemos ver, neste trabalho foi utilizado o modelo "spaCy", para a língua inglesa, no âmbito do processamento de linguagem natural, tendo sido aplicado um tokenizer e um featurizer também da spaCy, que permitem extrair os tokens e representar vetorialmente, respetivamente, o conteúdo introduzido.

```

pipeline:
  - name: "SpacyNLP"
    model: "en_core_web_md"
    case_sensitive: False
  - name: "SpacyTokenizer"
    intent_tokenization_flag: False
    intent_split_symbol: "_"
    token_pattern: None
  - name: "SpacyFeaturizer"
    pooling: "mean"
  - name: LanguageModelFeaturizer
    model_name: "bert"
    model_weights: "rasa/LaBSE"
    cache_dir: null
  - name: LexicalSyntacticFeaturizer
    case_sensitive: False
  - name: CountVectorsFeaturizer
    analyzer: char_wb
    min_ngram: 1
    max_ngram: 4
  - name: DIETClassifier
    entity_recognition: False
    constrain_similarities: true
    ambiguity_threshold: 0.2
  - name: FallbackClassifier
    threshold: 0.2

```

Figura 1: Pipeline de processamento

2.3 Utilização do Hugging Face

No âmbito de utilização da ferramenta Hugging Face, aplicou-se o modelo ”BERT”, especialmente adaptado para a arquitetura Rasa[2], no processamento das queries do utilizador, através do featurizer da spaCy.

Para além deste, foram usados mais 2 featurizers (”LexicalSyntacticFeaturizer” e ”CountVectorsFeaturizer”) para lidar com erros de escrita, por parte do utilizador. Por último foram utilizados 2 classificadores de intenção, que com base nos vetores previamente definidos, calculam qual a intenção expressa pelo utilizador (”DIETClassifier”). Caso se verifique ambiguidade na classificação, a intenção é classificada em fallback, desencadeando um pedido de reformulação da query (”FallbackClassifier”).

3 Diálogo

3.1 Interação inicial

Após a extração e classificação da intenção expressa pelo utilizador é necessário atuar sobre esta, de modo a estabelecer um diálogo. Para tal, uma das formas de estabelecer

reações a determinadas intenções, neste trabalho, foi o uso de regras, especificadas no documento "rules.yml".

Uma outra abordagem foi também o uso de histórias, definidas com recurso ao ficheiro "stories.yml". Estas duas formas de definir o discurso apresentam diferenças significativas, uma vez que as primeiras são desencadeadas sempre que determinada condição se verifique, e as últimas determinam os possíveis fluxos que a interação pode ter. Posto isto, recorreu-se a ambas as abordagens, de modo a que estas se complementem.

Uma das primeiras regras definidas neste projeto é relativa ao começo de um diálogo, que indica ao chatbot que deve responder com uma saudação, inquirindo o utilizador quanto ao seu estado de espírito atual, explicitada pela ação "utter_greet", sempre que o utilizador expresse uma salutação.

```
— rule: Say hi everytime the users says hi
  steps:
    — intent: greet
    — action: utter_greet
```

A resposta do utilizador poderá ser positiva ou negativa, dependendo do seu estado de espírito, pelo que o seguimento do discurso é captado por 3 possíveis histórias.

Numa situação normal, o utilizador responderá de forma positiva à questão anterior, desencadeando um ciclo que lhe permitirá fazer questões relacionadas com o sono. Este ciclo é marcado através do checkpoint "sleep_help".

```
— story: happy path 1
  steps:
    — intent: mood_great
    — action: utter_happy
    — action: utter_question_init
    — checkpoint: sleep_help
```

Contudo, é possível que a resposta do utilizador seja negativa. Para tal desenvolveram-se 2 histórias, nas quais o diálogo se pode enquadrar. Numa primeira abordagem, é apresentada uma frase motivadora ao utilizador, à qual este deve reagir. Caso se sinta melhor, o chatbot pergunta se tem alguma questão relacionada com o sono, desencadeando o fluxo de respostas às perguntas que o utilizador possa fazer.

```
- story: sad path 1
steps:
- intent: mood_unhappy
- action: utter_cheer_up
- action: utter_did_that_help
- intent: affirm
- action: utter_happy
- action: utter_question_init
- checkpoint: sleep_help
```

Caso o utilizador negue uma melhoria de estado após a frase motivadora, o ciclo também é ativado, contudo o chatbot não pergunta ao utilizador se este tem alguma questão, apenas avisa que ficará disponível para ajudar.

```
- story: sad path 2
steps:
- intent: mood_unhappy
- action: utter_cheer_up
- action: utter_did_that_help
- intent: deny
- action: utter_dismiss
- checkpoint: sleep_help
```

3.2 Interação pergunta-resposta

Após a etapa anterior o diálogo com o utilizador é direcionado para um ciclo de pergunta-resposta, conseguido através da utilização de checkpoints em Rasa. Deste modo o utilizador pode questionar o chatbot pela ordem de perguntas que bem entender, e obviamente, repetir alguma caso o deseje.

Deste modo, a primeira pergunta a que o Rasa pode responder diz respeito ao número de horas que o utilizador deve dormir, assumindo que este é um adulto.

A história que permite então responder a esta questão encontra-se abaixo descrita:

```
- story: wants_sleep_help_1
  steps:
    - checkpoint: sleep_help
    - intent: sleep_question_1
    - action: utter_response_1
    - action: utter_question
    - checkpoint: sleep_help
```

Como podemos ver, quando a intenção é classificada como "sleep_question_1" é desencadeada a resposta à mesma, enunciada de seguida:

- "Great question! As an adult you should sleep for about 7 to 8 hours per night, if you sleep any less than that for an extended period of time be aware you are putting your health at peril."^[3]

A segunda questão que foi inserida na base de conhecimento deste chatbot é referente às etapas do sono, sendo a história correspondente definida do mesmo modo que a anterior:

```
- story: wants_sleep_help_2
  steps:
    - checkpoint: sleep_help
    - intent: sleep_question_2
    - action: utter_response_2
    - action: utter_question
    - checkpoint: sleep_help
```

Quando o modelo identifica a pergunta como a questão número 2, é então desencadeada a seguinte resposta:

- "Sleep is actually a bit complex. Through a night sleep you go through 4 to 6 sleep cycles, which can be divided in light sleep, deep sleep and REM sleep, this last one being the one you get to experience vivid dreaming. Check my sources if you wanna know more!"^[4]

A terceira interrogação à qual este chatbot é capaz de responder direciona para as consequências, mentais e físicas, de não praticar uma boa higiene de sono, definida de modo similar à anterior:

```
– story: wants_sleep_help_3
  steps:
    – checkpoint: sleep_help
    – intent: sleep_question_3
    – action: utter_response_3
    – action: utter_question
    – checkpoint: sleep_help
```

Ao identificar que o utilizador pretende saber mais sobre este tópico, a resposta 3 é desencadeada, informando o utilizador das diversas consequências, tal como podemos ver:

- ”The consequences of not getting enough sleep can range from mental to physical. From a mental standpoint it will be increasingly difficult to pay attention, making decisions, dealing with emotions, right until depression, suicide and risk-taking behaviour. Lack of sleep can also lead you to microsleep. This also affects your driving skills, as much or more than alcohol, and can make you prone to make more mistakes in your line of work.”[5]

A quarta e última questão para a qual este chatbot está munido de resposta, advém de uma iniciativa, por parte do utilizador, de melhor compreender como pode melhorar o seu sono, sendo a estrutura da história correspondente similar às anteriores:

```
– story: wants_sleep_help_4
  steps:
    – checkpoint: sleep_help
    – intent: sleep_question_4
    – action: utter_response_4
    – action: utter_question
    – checkpoint: sleep_help
```

Após captar esta intenção, de saber como melhorar os seus hábitos de sono, o modelo é capaz de responder, utilizando a resposta 4, que apresenta diversas dicas para o efeito, tal como é possível verificar seguidamente:

- ”There are a couple tips I can give you, like going to bed and waking up at established hours; Exercising, though not close to bedtime; Avoiding caffeine, nicotine and alcohol; Trying to relax before going to bed; Avoiding screen time before going to bed; Taking your mind off falling asleep, as anxiety can contribute to insomnia, by reading a book for example.”[3]

3.3 Interações alternativas

Apesar de ser possível interagir em modo pergunta-resposta com o chatbot, não se pretende que este esteja limitado a esta função, pelo que foram definidas formas alternativas de interagir com este. Deste modo, o utilizador pode optar por não pretender questionar o robô, expressando esta vontade na resposta à pergunta inicial, que será captada pelo intent "not_eager_to_learn":

- "Is there any sleep related question you want to ask me about?

Deste modo, torna-se necessário definir um caminho que a história possa tomar nesta situação. Para isto foi desenvolvida uma história, que finda o ciclo de perguntas-respostas, introduzido pelo checkpoint, que podemos analisar de seguida:

```
— story: does_not_want_sleep_help
  steps:
    — checkpoint: sleep_help
    — intent: not_eager_to_learn
    — action: utter_dismiss
```

3.3.1 Factos de sono

De modo a introduzir uma outra componente didática, foi definido uma nova regra, apresentada de seguida, que permite ao utilizador obter nova informação sobre a forma de curiosidades, ou factos interessantes.

```
— rule: Tell the user a sleep related fact
  steps:
    — intent: sleep_facts
    — action: utter_sleep_facts
```

Os factos introduzidos na base de conhecimento são os que se seguem:

- "Did you know that your brain actually heals you when you are asleep? It does so by removing toxins that were produced during the day, pretty much like a kidney does. How fascinating is that?"^[5]
- "Many people 'sleep' on their sleep needs due to thinking they don't need as much! Not getting enough though will make you prone to various health problems, as heart disease and stroke, to dementia and obesity. Sleep is pretty much the elixir of life."^[3]

3.3.2 Fontes

Ao longo deste trabalho recorreu-se a diferentes fontes, de modo a obter informação fidedigna e de qualidade nas respostas fornecidas pelo chatbot desenvolvido. Num regime de transparência e de honestidade para com o utilizador, é essencial que este possa consultá-las, surgindo a necessidade de fornecer um método de consulta.

Por este motivo foi desenvolvida uma nova regra, que imprime as fontes utilizadas sempre que o utilizador inquirir o robô neste sentido, expressa da seguinte forma:

```
– rule: sources
steps:
– intent: want_sources
– action: utter_sources
```

3.3.3 Ajuda

De um ponto de vista técnico podemos saber quais funcionalidades esperar do chatbot, e como as explorar, de forma bastante simples, bastando para isso apenas consultar o código-fonte, contudo isto não se verifica do lado do utilizador.

Tendo este ponto de vista subjacente, desenvolveu-se uma regra que permite ao utilizador obter ajuda, relativamente às funcionalidades disponíveis, de forma simples e concreta, simplificando a utilização do chatbot.

```
– rule: help
steps:
– intent: need_help
– action: utter_help
```

Como podemos ver, um simples pedido de ajuda, como por exemplo "help", desencadeia uma resposta informativa, aqui apresentada:

- "I can help you with sleep related questions, like the amount of sleep you need; How to improve your sleep; The different stages of sleep; The problems that might arise from poor sleep hygiene. I can also present you with some facts you might find interesting!"

3.3.4 Interações finais

Existem duas formas do utilizador terminar o diálogo com o chatbot, permitidas pelas regras que serão de seguida apresentadas.

A primeira verifica-se aquando o modelo de linguagem identifica um agradecimento por parte do utilizador, desencadeando uma resposta cordial. A implementação da regra apresenta-se de seguida:

```
- rule: you're-welcome
-- steps:
--- intent: - thanking
--- action: - utter>Welcome
```

Por outro lado, e sendo uma forma mais direta de finalização de diálogo, o modelo é capaz de identificar despedidas, e responder também, cordialmente, despedindo-se. Já esta regra traduz-se da seguinte forma:

```
- rule: Say goodbye anytime the user says goodbye
  steps:
    - intent: goodbye
    - action: utter_goodbye
```

3.3.5 Aspetos legais

Por se tratar de um chatbot meramente informativo, este não recolhe informações pessoais do utilizador, pelo que não se aplica as questões de dados pessoais.

É, contudo, um direito do utilizador saber que está a interagir com um robô, obrigação cumprida através da regra seguidamente apresentada, em que, quando o utilizador questiona sobre a natureza do chatbot, este afirma ser um robô.

```
- rule: Say 'I-am-a-bot' anytime the user challenges
  steps:
    - intent: bot_challenge
    - action: utter_iamabot
```

Por se tratar de uma aplicação de cariz unicamente académico, julga-se que esta medida é suficiente, contudo, em contexto real, o chatbot poderia, de cada vez que é iniciada uma conversa, enunciar que se trata de um chatbot.

3.4 Lidar com a incerteza e desconhecido

Na secção 2 abordámos as diferentes intenções que o chatbot desenvolvido era capaz de identificar, e abordou-se a forma como o pipeline foi desenhado para este efeito.

Vimos ainda que, em situações em que a confiança de classificação era baixa, o classificador "FallbackClassifier" desencadeava uma ação de reformulação.

De modo a tornar isto possível, foi necessário contudo, definir uma regra que desencadeasse esta ação, aquando uma "nlu_fallback":

```
— rule: Ask the user to rephrase whenever they send  
a message with low NLU confidence  
steps:  
— intent: nlu_fallback  
— action: utter_please_rephrase
```

Ao longo da produção deste trabalho, tornou-se rapidamente claro, que seria impossível lidar com todos os desvios do diálogo que o utilizador poderia tomar, contudo foram impostas algumas 'defesas' contra este tipo de situações, de modo a não quebrar imediatamente a lógica do diálogo. A definição do intent "out-of-scope", previamente abordado, foi uma destas primeiras 'defesas', em conjunto com a regra apresentada em baixo:

```
— rule: out-of-scope  
steps:  
— intent: out-of-scope  
— action: utter_out_of_scope
```

Apesar de moderadamente eficaz, este mecanismo está dependente dos casos de treino, que por sua vez é relativamente limitado, quando comparado com a míriade de assuntos para o qual este chatbot não sabe, nem deve, responder.

4 Políticas

Previvamente foram abordados questões relativas à estruturação do diálogo, nomeadamente quais as intenções na base de conhecimento do chatbot e ainda como reagir a estas intenções, bem como algumas histórias e regras que unificam tudo isto.

Uma questão que ainda não foi respondida contudo é, como é que o chatbot 'sabe' o que tem de fazer?

Voltando ao ficheiro "config.yml", podemos definir uma secção de 'policies', que determinam quais os fatores, e qual a sua prioridade, na tomada de decisão por parte do chatbot.

Estas políticas podem ser divididas em 2 tipos, as baseadas em machine learning, e as baseadas em regras.

4.1 RulePolicy

A política com maior prioridade definida neste trabalho trata-se da ”RulePolicy”. Esta política, obviamente baseada em regras, faz previsões com base nas regras definidas pelo utilizador, pelo que é essencial para que estas funcionem.

4.2 MemoizationPolicy

Descendo em nível de prioridade e entrando no campo das políticas baseadas em machine learning, temos a política ”MemoizationPolicy”. Esta política é a que permite ao modelo ‘lembra-se’ das histórias com que foi treinado, tornando possível as interações que caem no espetro definido por estas.

4.3 UnexpecTEDIntentPolicy

Num nível de prioridade abaixo da política anterior incluiu-se a ”UnexpecTEDIntentPolicy”. Esta política funciona como um complemento para a que será apresentada de seguida, contudo parece ser interessante, na medida em que permite melhor reagir a uma ação, por parte do utilizador, que não seria esperada naquele ponto da conversa, adicionando fluidez e resolvendo a problemática de abandono de história.

4.4 TEDPolicy

No nível mais baixo de prioridade temos a política ”TEDPolicy”. Esta política é responsável pela previsão da próxima ação a tomar pelo chatbot, tendo sido definido um histórico máximo de 8 interações.

```
policies:
- name: MemoizationPolicy
  max_history: 4
- name: RulePolicy
  core_fallback_threshold: 0.2
  core_fallback_action_name: "action_default_fallback"
  enable_fallback_prediction: True
- name: UnexpecTEDIntentPolicy
  epochs: 50
- name: TEDPolicy
  max_history: 8
  epochs: 150
  constrain_similarities: true
```

Figura 2: Políticas e parâmetros aplicados”

Com estas políticas é então possível dotar o robô desenvolvido de alguma percepção e inteligência, definindo quais as heurísticas que devem ser utilizadas na tomada de decisão, sobre as quais o modelo atuará.

5 Avaliação de experiência de utilizadores

Uma das principais recomendações da documentação, fornecida pela Rasa, é testar os chatbots desenvolvidos com diferentes utilizadores, o mais cedo possível. Apesar de limitado, foram realizados alguns testes, com resultados pouco promissores, contudo estes serviram para melhor entender os pontos de falha do chatbot desenvolvido, e melhor adaptá-lo para lidar com as situações encontradas.

Reconhece-se, contudo, que este é o maior ponto de falha do trabalho desenvolvido, devido à quantidade inicial de utilizadores de teste, que apesar de ter ajudado a melhorar significativamente o trabalho. Um maior número de utilizadores traria, possivelmente, mais problemas, que simplesmente não são conhecidos, e para os quais este chatbot não apresenta solução.

De seguida vamos analisar uma interação particular, obtida nesta etapa, que captou diversos problemas na evolução do diálogo.

```
— user: hi
— rasa: Hi! I was wondering when you would show up.
How have you been?
— user: bad
— rasa: Hapiness can be found in the darkest of times ,
if one only remembers to turn on the light . — J.K. Rowling .
— rasa: Did i manage to make it better?
— user: no
— rasa: I 'm- sorry -to -hear -that ! -I 'll be here
if there is anything you need.
— user: help
— rasa: I 'm- sorry , -I -didn 't quite understand that .
Can you rephrase your question please?
— user: i need help
— rasa: I 'm- sorry , -can -you -write -your -question
any -other -way?
— user: - i -wnant -to -sleep -better
— rasa: - I 'm sorry , can you write your question
any other way?
— user: need to sleep better
— rasa: I 'm- sorry , -I -didn 't quite understand that .
Can you rephrase your question please?
```

Este feedback foi crucial para o diagnóstico de alguns problemas iniciais do chatbot inicial.

O primeiro erro identificado foi a facilidade com que o utilizador quebrou a história, com a interação inicial. Neste ponto de desenvolvimento, a história "sad path 2" ainda não tinha sido implementada, pois até este momento os utilizadores eram bem intencionados.

Com esta simples implementação, este problema ficou resolvido, sem impedir o desenvolvimento futuro da conversa.

O erro seguinte surgiu de forma mais inusitada, a não identificação de um pedido de ajuda, inesperado, pois a resposta tinha sido implementada através de uma regra, que não dependia do momento ou história em que o utilizador se podia encontrar.

Deste modo, concluiu-se que o erro estava na forma como o modelo identificava a intenção de "help".

Após nova análise, verificou-se que um dos exemplos de treino de outra intenção continha a palavra help, e sendo esta a palavra chave na anterior, esta gerava confusão na decisão, manifestando-se através de baixos níveis de certeza de classificação.

Após uma restruturação dos exemplos de treino, este problema ficou também resolvido.

Ao corrigir o primeiro problema, a não identificação da questão ficou também resolvida, contudo, o erro ortográfico introduzido contribuiu para a introdução do "CountVectorsFeaturizer", responsável por extrair as palavras sob a forma de n-gramas, com tamanho definido entre 1 e 4, para uma bag of words, e ainda o "LexicalSyntacticFeaturizer", de modo a agrupar palavras lexical e semânticamente próximas. Estes featurizers ajudam na melhor compreensão, e comparação das frases introduzidas pelo utilizador.

Destes testes resultou também uma alteração às mensagens de pedido de reformulação, na forma de um direcionamento para a consulta das fontes utilizadas em caso de dúvida permanente.

É de notar a influência significativamente positiva do modelo BERT na obtenção de melhores níveis de confiança na escolha da intenção.

6 Conclusão e trabalho futuro

Em suma, deste trabalho resulta um chatbot especialista na área médica do sono, capaz de responder a 4 diferentes perguntas, e algumas formas variantes de as colocar, e ainda enunciar 2 factos relevantes sobre a área em análise.

No desempenhar da sua função, é capaz de lidar com algumas nuances da interação humana, de forma cordial e enunciar as fontes utilizadas na obtenção da informação exposta.

Prevê-se também um chatbot com domínio de resposta fechado, não respondendo a questões fora deste, o que é desejável em contexto de especialista de saúde.

Ainda mais foram cumpridos os requisitos legais de implementação apresentados.

Tendo sido construído de forma modular, a inclusão de mais perguntas seria algo desejável numa implementação futura, bem como a inclusão de mais factos e diferentes formas de formular as respostas, de modo a que estas sejam mais naturais. Uma melhoria, também já mencionada ao longo deste relatório, seria uma maior divulgação do próprio chatbot, a fim de captar mais exemplos de formulação, e melhor afinar os diversos parâmetros do modelo.

Posto isto, o autor crê ter atingido todos os objetivos propostos, apresentando uma solução robusta e capaz ao problema proposto, tendo sido claramente identificados os pontos de melhoria para trabalho futuro.

Referências

- [1] Rasa Open Source Docs. <https://rasa.com/docs/rasa/>. Accessado: 17-05-2024
- [2] rasa/LaBSE. <https://huggingface.co/rasa/LaBSE>. Accessado: 17-05-2024
- [3] The George Washington University Hospital. <https://www.gwhospital.com/frequently-asked-questions-about-sleep>. Accessado: 17-05-2024
- [4] Sleep Foundation. <https://www.sleepfoundation.org/stages-of-sleep>. Accessado: 17-05-2024
- [5] NIH - National Heart, Lung, and Blood Institute. <https://www.nhlbi.nih.gov/health/sleep-deprivation/health-effects>. Accessado: 17-05-2024