Língua Natural

Instituto Superior Técnico - Campus TagusPark

Mini Projecto 2 (2015/2016)

Número do grupo: G03

Nome: Bruno Alexandre Pires Henriques **Número:** 72913

Nome: Tiago Manuel Ferrão dos Santos **Número:** 72960

1. Introdução

Na primeira parte do projecto foi desenvolvida a arquitectura base para um sistema de *Questions and Answers,* no entanto o seu desempenho não era ideal, pois não tinha em conta a semelhança entre as frases. De forma a melhorar a *accuracy* do sistema foram desenvolvidas técnicas para verificar semelhanças entre *User Input* e *Triggers* e semelhanças entre respostas.

Nas secções seguintes vamos apresentar a arquitectura do sistema, as estratégias utilizadas, a discussão dos resultados obtidos e por fim apresentar como o seria possível melhorar o sistema.

2. Arquitectura

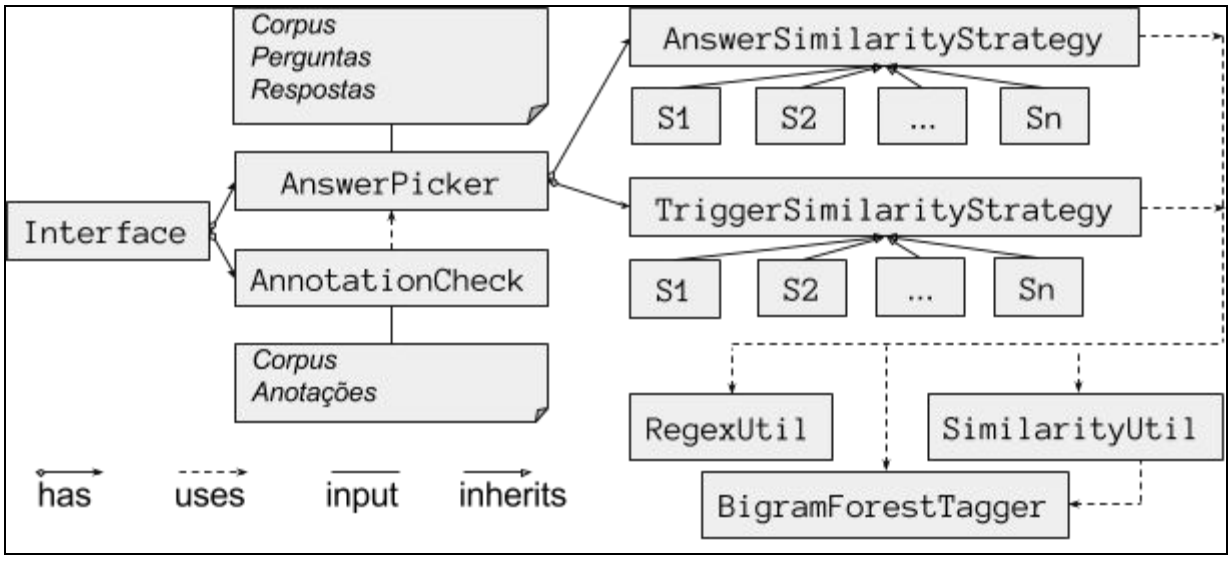

Figura 1 - Arquitectura simplificada da aplicação

À semelhança da primeira parte do projecto, o utilizador interage com a função `myAvalia` que retorna a *accuracy* do sistema dada uma lista de perguntas. A sua assinatura, foi, no entanto, modificada para receber a estratégia utilizada para verificar se o *User Input* e o *Trigger* eram semelhantes o suficiente (`TriggerSimilarityStrategy`) e a estratégia utilizada para verificar o mesmo mas para duas respostas (`AnswerSimilarityStrategy`). Esta opção permitiu desenvolver novas estratégias sem alterar o núcleo do algoritmo e permitiu testar qual seria a melhor combinação que tornasse o sistema mais preciso.

`RegexUtil` é responsável por fazer a normalização das frases. `SimilarityUtil` contém as medidas de semelhança. Por fim, a classe `BigramForestTagger` é o responsável por fazer a análise morfológico usando todo o *Corpus* Floresta.

2.1 Medidas de distância

2.1.1. *Jaccard* e *Dice*

Na língua Portuguesa, frases como "Onde queres ir tomar café?" e "Queres ir tomar café onde?" são perguntas, que apesar da ordem das suas palavras serem diferentes, têm o mesmo significado. Nesse sentido, como objectivo de identificar estes padrões, foram desenvolvidos o *Jaccard* e o *Dice* como medidas de distância.

O *Jaccard* é uma medida de distância que permite comparar frases como conjuntos, ignorando a ordem e a frequência das palavras que contém. O *Dice* apresenta as mesmas vantagens mas atribui menos peso à diferença de comprimento entre frases, como tal, palavras adicionais que não alteram o significado da pergunta (podem não ser *stop-words*) não terão tanto impacto como teriam no *Jaccard*.

$Jaccard(A,\ B) = \frac{count(A \cap B)}{count(AUB) - count(A \cap B)}$	$Dice(A,\ B) = 2 \times \frac{count(A \cap B)}{count(A) + count(B)}$

2.1.2. Braccard

Braccard (nomeado pelos autores) é uma medida de distância baseada em *Jaccard* com uma componente adicional morfológica. Em *Braccard*, assumimos que o conjunto não-intersecção contém informação morfológica relevante, pois as palavras da mesma categoria morfológica podem desempenhar a mesma função semântica. Por exemplo, o conjunto de não-intersecção entre as frases "Queres ir tomar café, João?" e "Queres ir tomar café, Maria?" é {"João", "Maria"} que são ambos nomes próprios e desempenham o mesmo papel. Seguindo esta lógica substituímos cada palavra nos dois conjuntos de não-intersecção pela sua *tag* e calculamos *Jaccard* novamente, adicionando um peso (entre 0 e 1) a essa componente, sendo que a fórmula final é:

$$Braccard(A,\ B) = \frac{count(A \cap B)}{count(AUB) - count(A \cap B)} + \frac{weight * count(Tag(A - A \cap B) \cap Tag(B - A \cap B))}{count(AUB) - count(A \cap B)}$$

2.1.3. MED

O *Minimum Edit Distance* é uma medida que ao contrário de *Jaccard* e *Dice*, analisa as frases como sequências e não como conjuntos de palavras identificado o número mínimo de transformações: inserções substituições ou remoções. Nesta parte do projecto foi considerado que cada uma todas estas transformações têm o mesmo peso (unitário).

2.1.4. YesNoSimilar (apenas para respostas)

Perguntas fechadas são bastante comuns, i.e, perguntas cujo interrogador espera uma resposta curta que frequentemente é apenas uma palavra. Nesse sentido, por análise do *Corpus* de desenvolvimento, denotámos que perguntas cuja resposta é “sim” ou “não” são frequentes (e.g. “Tens filhos?”). Por consequente colocámos como hipótese que a essência da resposta estivesse contida nestes mesmo advérbios. Desta forma, duas respostas afirmativas ou duas respostas negativas serão semelhantes segundo esta heurística.

O peso dado aos mesmos é arbitrário e o restante peso é calculado com outra medida, como por exemplo, *Jaccard* ou *Dice* como demonstrado no pseudo-código seguinte:

```
YesNoSimilar(A, B, weight, f) =
     if ('sim' in A and 'sim' in B) or ('não' in A and 'não' in B)
          return weight + (1-weight) * f(A,B)
     else
          return f(A,B)
```

3. Pré-processamento

De forma a facilitar o processamento e uniformizar as frases (fruto da expressividade da língua Portuguesa) desenvolvemos várias estratégias de pré-processamento:

A. *Stemming*: reduzir a palavra ao seu significado principal;
B. Remoção de *stop-words*: remoção de palavras que não alteram o significado da frase;
C. Remoção de frases não interrogativas: no *Corpus* de desenvolvimento frases como “Óptimo. Dás me o teu telefone?” passam para “Dás me o teu telefone?”;
D. Normalização das frases: Substituir caracteres acentuados, seguido de *lowercasing*, seguido de remover a pontuação e por fim `strip`;
E. Filtrar nomes comuns, nomes próprios, proposições, artigos, pronomes (pessoais, determinativos e independentes). Este filtro é apenas aplicado nos *Triggers*, pois, existem perguntas cujo nome próprio é relevante, e.g., “Quem matou a Maria?”.

	Medida de semelhança	Filtros aplicados
Triggers	Jaccard, Dice, MED	C -> E ->B -> A -> D
	Braccard	B
Answers	Jaccard, Dice, MED, YesNoSimilar	B -> A -> D
	Braccard	B

4. Setup Experimental

Tal como a primeira parte do projecto, foram utilizados 2 *Corpora*: um com conjunto {*User Input*, *Trigger*, *Answers}* não anotado e outro com o *User Input* e as *Answers* anotadas de acordo com a sua plausibilidade (*'y'* como correto, *'n'* como incorreto ou *'m'* como correto dependendo do contexto). De seguida, foi retirados do primeiro *Corpus* todos as perguntas aceites pelo sistema e dividiu-se aleatoriamente em dois conjuntos: um com o conjunto com 198 perguntas para desenvolvimento (85%) e outro com 35 perguntas para teste (15%).

A partir do conjunto de medidas de semelhança entre *Triggers* (*Jaccard*, *Dice*, *MED*, *Braccard*) e entre answers (*Jaccard*, *Dice*, *MED*, *Braccard*, *YesNoSimilar Jaccard* e *YesNoSimilar Dice*) foram realizadas todas as combinações possíveis e as suas variações com e sem o pré-processamento para diversos limites inferiores de semelhança aceitável e foi calculada a *accuracy* do sistema cujos resultados resumidos encontram-se na secção seguinte.

5. Resultados

As melhores estratégias encontram-se ilustradas na tabela seguinte:

	Input/Trigger	Strategy Answers	Devel Accuracy	Test Accuracy
A	Jaccard - T=0.25, F=False	Braccard(W=0.25) - T=0.25, F=False	45.685%	31.429%
B	Jaccard - T=0.25, F=False	Braccard(W=0.50) - T=0.50, F=True	45.178%	37.143%
C	Jaccard - T=0.50, F=False	MED - MV=1, F=False	44.670%	28.571%
D	Braccard(W=0.25) - T=0.25, F=True	MED - MV=1, F=False	44.162%	37.143%
E	Braccard(W=0.50) - T=0.25 F=True	YesNoSimilar(W=0.50, dice_sentence) - T=0.75, F-True	44.162%	42.857%
F	Jaccard - T=0.50, F=False	YesNoSimilar(W=0.50, dice_sentence) - T=0.75, F-True	44.162%	28.571%
G	MED - MV=2, F=False	Jaccard - T=0.50, F=False	43.147%	31.429%
H	MED - MV=2, F=False	Dice - T=0.75, F=True	43.147%	25.714%
	T = Threshold Minimum Value**, F =** Filter**, MV =** Threshold Maximum Value**, W =** Weight			

A partir da lista de perguntas para desenvolvimento foi verificado que a estratégia **A** foi a que obteve melhores resultados, no entanto, a mesma estratégia não teve o mesmo sucesso quando aplicado à lista de perguntas para teste. Adicionalmente, verificou-se que o facto de aplicarmos filtros nas respostas *a priori* reduziu a *accuracy* em 0.5%.

A combinação de estratégias que teve mais *accuracy* para o conjunto de teste foi *Braccard* (com remoção das *stop-words*) nas perguntas e *YesOrNoSimilar* (aplicando *Dice* e os filtros) nas respostas e verificou-se, tanto na lista de perguntas para desenvolvimento (e nos testes) que, em média, os filtros pioraram os resultados.

Braccard, tendo em conta o desempenho no conjunto de desenvolvimento e o as perguntas presentes no conjunto de teste, faz sentido que seja adequado. Muitos *User Inputs* no conjunto de teste são semelhantes aos *Triggers* exceptuando em uma palavra que é da mesma classe morfológica que as restantes e consequentemente consegue encontrar um *Trigger* facilmente. Mas com o *threshold* utilizado é relativamente simples encontrar semelhanças visto que basta as frases serem 25% semelhantes para serem consideradas

semelhantes. Portanto é possível o conjunto de teste de tamanho reduzido tenha *Triggers* com palavras chave, como “Onde”, “Como”, “Quando” que estejam presentes no *User Input*. Regra geral, estas perguntas podem ser respondidas da mesma forma, sem saber o resto da frase e portanto gerariam respostas plausíveis.

O *YesOrNoSimilar* tem bons resultados devido a agrupar respostas “Não sei”, “Não me lembro” “Não” que são frequentes. São respostas de significado semelhante, poucas são as respostas onde uma não pode ser substituída pela outra mantendo plausibilidade.

Os resultados não foram melhores devido à qualidade das estratégias, que poderiam ser melhoradas, e do *Corpus* que para muitas perguntas não tinha uma resposta válida, ou mesmo tendo uma resposta válida as respostas não tinha um *Trigger* semelhante que permitisse a sua escolha.

6. Conclusão e Trabalho Futuro

Por análise dos resultados apresentados, concluímos que a utilização do *Braccard* (filtrando as *stop-words*) para comparação entre *User Input* e *Triggers* e a estratégia *YesNoSimilar* (com filtro) para as respostas permite melhores resultados, pois os valores de *accuracy* foram dos melhores na lista de perguntas de desenvolvimento e a melhor na lista para testes. No entanto, este teste é limitado e dever-se-ia confirmar no futuro a consistência dos resultados obtidos com *k-fold cross validation* aplicando a média dos resultados para outros conjuntos de perguntas aleatórias. É de salientar que os autores reconhecem que existe espaço para melhoramentos das técnicas desenvolvidas, pois tratam-se acima de tudo heurísticas desenvolvidas dado o *Corpus* de desenvolvimento.

Por exemplo, *Braccard* compara os conjuntos de não-intersecção a partir da sua tag, o que significa que em frases completamente distintas (em termos de palavras constituintes) poderá existir algum grau de semelhança. E.x “Onde está a banana?” e “Como é esta semana?” serão semelhantes pois têm um conjunto de *tags* idêntico (‘Nome’, ‘verbo’, ‘determinante’, ‘nome’). Uma das possíveis soluções para este problema é atribuir pesos às *tags*, permitindo então que seja atribuída uma maior/menor importância a determinadas categorias morfológicas. Ainda em *Braccard*, sugere-se uma alternativa *Brice* que usa o *Dice* como medida de semelhança base em vez do *Jaccard.*

A estratégia de semelhança para respostas *YesNoSimilar* também poderia ser melhorada através da utilização de expressões sinônimas, i.e. poderia passar em ter em conta, como por exemplo “ok” para respostas afirmativas ou “nem por isso” para respostas negativas.

Tal como referido, os resultados foram em média piores quando se aplicava filtros. Deste modo, os autores julgam que dever-se-ia tornar o filtro de frases não interrogativas mais robusto e explorar outras categorias durante a filtragem de categorias morfológicas nos *Triggers* e tentar aplicar variações do mesmo filtro nas respostas e estudar o seu impacto.

Por fim, No algoritmo base seria desejável obter uma reposta em todas as situações. Nesse sentido, quando não é encontrado um *Trigger* semelhante para um dado limite inferior de semelhança poder-se-ia diminuir esse limite e procurar novamente, garantindo assim pelo menos uma resposta para todas as perguntas.