Introdução à Inteligência Artificial

Teste individual prático nº 2
Extensões a um módulo de redes semânticas

Ano Lectivo de 2013/2014

7 de Novembro de 2013

I Observações importantes

1. Este trabalho de casa deverá ser entregue no prazo de 24 horas após a publicação deste enunciado
2. Em cada módulo pedido, inclua um comentário com o seu nome e número mecanográfico
3. Pode discutir o enunciado com colegas, mas não pode copiar programas, ou partes de programas, qualquer que seja a sua origem
4. Se discutir o trabalho com colegas, inclua um comentário com o nome e número mecanográfico desses colegas
5. Se recorrer a outras fontes, identifique essas fontes também
6. Todo o código submetido deverá ser original; embora confiando que a maioria dos alunos fará isso, serão usadas ferramentas de detecção de copianço
7. Alunos que recorrerem a copianço terão os seus trabalhos anulados
8. Os trabalhos poderão ser entregues para além do prazo de 24 horas, mas serão penalizados em 5% por cada hora adicional
9. Os programas serão avaliados tento a conta a correcção, completude, estilo e evidência de trabalho autónomo (independente)

II Exercícios

Em anexo a este enunciado, pode encontrar o módulo semantic_network, que usou nas aulas práticas. Para a resolução dos exercícios, deve criar um novo módulo mysemnet, e nesse módulo

deve criar uma classe MySN como classe derivada de SemanticNetwork. Nos exercícios propostos, serão pedidos novos métodos a incluir na classe derivada. Em anexo, pode ainda encontrar o módulo mysn_example, com uma rede semântica já criada. Os exemplos dados abaixo, baseiam-se neste exemplo.

1. Desenvolva uma nova função most_frequent_triple () na classe MySN, a qual deve determinar o triplo $(entity1, relation, entity2)$ correspondente à relação mais frequentemente declarada. A função retorna um par com o triplo mais frequente e o número de declarações desse triplo.

   Exemplo:

   ```
   >>> z.most_frequent_triple()
   (('socrates', 'professor', 'filosofia'), 4)
   >>>
   ```

2. Desenvolva uma nova função query_assoc() na classe MySN, a qual permite fazer consultas de valores de associações, com herança e cancelamento de herança. Assumindo que as associações admitem apenas um valor, a existência de valores declarados diferentes deve ser tratada determinando e retornando o valor mais frequente. A função deve suportar multipla herança, ou seja, uma associação pode herdar de dois predecessores imediatos. A função recebe como entrada uma entidade e (o nome de) uma associação, e retorna uma lista contendo o valor ou valores encontrados.

   Exemplos:

   ```
   >>> z.query_assoc('socrates','professor')
   [('socrates', 'filosofia', 0.80)]
   >>> z.query_assoc('socrates','tem_cerebro')
   [('homem', 'sim', 0.67), ('filosofo', 'sim', 1.00)]
   >>>
   ```

3. O módulo semantic_network, que já conhece bem, exporta as classes Member, Subtype e Association, para representar diferentes relações semânticas. Tal como assumido no exercício anterior, as associações aceitam apenas um valor possível e, existindo vários declarados, conta o mais frequente.

   Ha, no entanto, outros tipos de associações. Por exemplo, a associação gosta pode admitir vários valores (pode-se gostar de peixe e de carne). Já no caso de associações com valor numérico (por exemplo altura ou peso), a existência de vários valores declarados pode ser tradada como uma distribuição, tomando-se a média desses valores como uma boa aproximação à verdade.

   Para testar as duas alíneas seguintes, descomente as declarações que estão comentadas no módulo mysn_example.

   (a) Desenvolva duas classes derivadas da classe Relation para representar associações com múltiplos valores (classe AssocSome) e associações com valores numéricos (classe AssocNum).

   (b) Desenvolva uma nova função query_local_assoc () na classe MySN, a qual permite fazer consultas de valores das associações locais de uma dada entidade, devendo processar os diferentes tipos de associações da seguinte forma:

   - Association -Devolve um par $(val, freq)$, em que val é o valor local mais frequente, e $freq$ é frequência (percentagem) com que ocorre.

- AssocSome - Devolve uma lista de pares $(val, freq)$ com os valores locais mais frequentes e respectivas frequências. Na selecção dos valores mais frequentes, os valores são acrescentados à lista por ordem decrescente da sua frequência, até a soma das frequências atingir um valor não inferior a 0.75.
- AssocNum - Devolve a média dos valores locais.

A função recebe com entrada um entidade e (o nome de) uma associação, e devolve o resultado tal como descrito acima.

Exemplos:

```
>>> z.query_local_assoc('socrates','professor')
('filosofia', 0.80)
>>> z.query_local_assoc('homem','gosta')
[('carne', 0.40), ('peixe', 0.40)]
>>> z.query_local_assoc('socrates','peso')
77.5
>>>
```