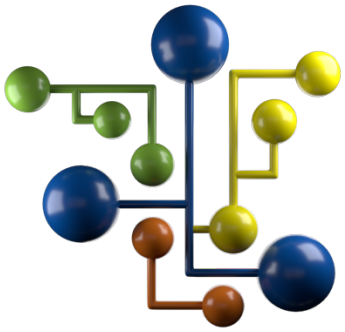

Data Science Academy

www.datascienceacademy.com.br

Processamento de Linguagem Natural

Modelo n-grama e Cadeia de Markov

Em última análise, um texto escrito é composto por **caracteres** — letras, dígitos, pontuação e espaços (e caracteres mais exóticos em alguns outros idiomas). Assim, um dos modelos de linguagem mais simples é uma distribuição de probabilidade sobre sequências de caracteres. Escrevemos P(c1:*n*) para a probabilidade de uma sequência de *N* caracteres, de *c*1 até *cn*.

Uma sequência de símbolos escritos de comprimento n é chamado de n-grama (da raiz grega para escrita ou letras), com o caso especial de 1-grama para "unigrama", 2-gramas para "bigrama" e 3-gramas pra "trigrama". Um modelo de distribuição de probabilidade de n sequências de letras é então chamado de modelo de n-grama (mas, cuidado: podemos ter modelos de n-gramas com sequências de palavras, sílabas ou outras unidades, não apenas de caracteres). Um modelo de n-grama é definido como uma cadeia de Markov de ordem n − 1. Em uma cadeia de Markov, há a probabilidade do caractere depender apenas dos caracteres imediatamente anteriores e não de qualquer outro caractere. Assim, em um modelo de trigrama (cadeia de Markov de ordem 2) temos esta fórmula:

$$P(c_i \mid c_{1:i-1}) = P(c_i \mid c_{i-2:i-1})$$

Podemos definir a probabilidade de uma sequência de caracteres P(c1: N) sob o modelo do trigrama primeiro por fatoração com a regra de cadeia e, em seguida, utilizando a hipótese de Markov. Aqui no curso de Processamento de linguagem natural faremos a implementação usando linguagem de programação Python e pacotes e funções prontos, sem a necessidade de implementar este tipo de fórmula manualmente!

$$P(c_{1:N}) = \prod_{i=1}^{N} P(c_i \mid c_{1:i-1}) = \prod_{i=1}^{N} P(c_i \mid c_{i-2:i-1})$$

Para um modelo de caracteres de trigrama em uma linguagem com 100 caracteres, P(Ci | Ci–2:i–1) tem um milhão de entradas e pode ser estimado com precisão, contando as sequências de caracteres em um corpo de texto de 10 milhões de caracteres ou mais. Denominaremos corpus (plural corpora) um corpo de texto, da palavra latina para corpo.

O que podemos fazer com os modelos de caracteres de n-gramas? Uma tarefa para a qual eles estão bem adaptados é a identificação de linguagem: dado um texto, determine qual linguagem natural está escrita nele, o que pode ser usado por exemplo para tradução automática de textos. Essa é uma tarefa relativamente fácil, mesmo com textos curtos, como “Hello, world” ou “Eu quero viajar”; é fácil identificar o primeiro como inglês e o segundo como português. Os sistemas de computador identificam os idiomas com mais de 99% de precisão; ocasionalmente é confuso identificar idiomas que estão intimamente relacionados, tais como sueco e norueguês.

Uma abordagem para identificação de idioma é primeiro construir um modelo de caracteres de trigrama de cada idioma candidato, P(ci | ci–2:i–1,ℓ), onde a variável ℓ estende-se entre os idiomas. Para cada ℓ é construído um modelo através da contagem de trigramas em um corpus dessa língua (são necessários cerca de 100.000 caracteres de cada língua).

Isso nos dá um modelo de P(Texto | Linguagem), mas, dado o texto, queremos selecionar o idioma mais provável, então aplicamos a regra de Bayes seguida pela suposição de Markov para obter o idioma mais provável.

$$\begin{aligned}\ell^* &= \operatorname*{argmax}_{\ell} P(\ell \mid c_{1:N}) \\ &= \operatorname*{argmax}_{\ell} P(\ell) P(c_{1:N} \mid \ell) \\ &= \operatorname*{argmax}_{\ell} P(\ell) \prod_{i=1}^{N} P(c_i \mid c_{i-2:i-1}, \ell)\end{aligned}$$

O modelo de trigrama pode ser aprendido a partir de um corpus, mas, e sobre a probabilidade anterior P(l)? Podemos ter uma estimativa desses valores; por exemplo, se estivermos selecionando uma página aleatória da Web, sabemos que o inglês é o idioma mais provável e que a probabilidade de macedônio é inferior a 1%. O número exato que selecionamos para essas medidas a priori não é crítico porque o modelo de trigrama normalmente seleciona uma linguagem que será várias ordens de magnitude mais provável que qualquer outra.

Outras tarefas para modelos de caracteres incluem a correção de ortografia, classificação de gênero e o chamado reconhecimento de entidade. A classificação de gênero significa decidir se um texto é uma nova história, um documento legal, um artigo científico etc. Embora muitas características ajudem a fazer essa classificação, o resultado da pontuação e outros caracteres n-grama característicos podem contribuir muito. O objetivo da tarefa denominada reconhecimento de entidade é o de encontrar os nomes das coisas em um documento e decidir a qual classe pertencem. Por exemplo, no texto “Foi prescrito aspirina para o Sr. Morais”, devemos reconhecer que “o Sr. Morais” é o nome de uma pessoa e “Aspirina” é o nome de um medicamento. Os modelos de nível de caracteres são bons para essa tarefa.

A principal complicação de modelos n-grama é que o corpus de treinamento fornece apenas uma estimativa da distribuição de probabilidade verdadeira. Para sequências de caracteres comuns, como “ht” qualquer corpus em inglês dará uma boa estimativa. Por outro lado, “ht” não é muito comum — nenhuma palavra do dicionário começa com ht. É provável que a sequência tivesse uma contagem de zero em um corpus de treinamento de padrão inglês. Isso significa que se deve atribuir P(“ht”) = 0? Se fizermos isso, o texto “**The program issues an http request**” teria uma probabilidade de inglês zero, o que estaria errado. Temos um problema de generalização: queremos que nossos modelos de linguagem generalizem bem os textos que ainda não tenham visto. Apenas porque nunca vimos “ht” antes não significa que nosso modelo deva afirmar que é impossível. Assim, vamos ajustar o nosso modelo de linguagem de forma que será atribuída uma pequena probabilidade diferente de zero às sequências que tiverem resultado zero no corpus de treinamento (e os outros resultados serão ajustados ligeiramente para baixo de modo que a probabilidade ainda some 1).

O processo de ajustar o resultado da probabilidade de baixa frequência é chamado de alisamento. O tipo mais simples de alisamento foi sugerido por Pierre-Simon Laplace, no século XVIII: ele disse que, na falta de mais informações, se uma variável aleatória booleana X for falsa em todas as n observações até agora, a estimativa de P(X = verdadeiro) deve ser 1/(n +2). Isto é, supõe-se que, com mais duas tentativas, exista a chance de ser um verdadeiro e outro falso. O alisamento de Laplace (também chamado de alisamento exponencial) é um passo na direção certa, mas tem desempenho relativamente fraco. Uma abordagem melhor é o modelo de recuo (backoff), em que começamos por estimar o resultado de n-grama, mas para qualquer sequência particular que tenha resultado baixo (ou zero), recuamos para (n − 1) gramas. Caso você tenha feito a Formação Cientista de Dados, você deve se lembrar do alisamento de Laplace, que aplicamos em algumas oportunidades e estudamos no curso de Machine Learning.

A interpolação linear com alisamento é um modelo de recuo que combina modelos trigrama, bigrama e unigrama por interpolação linear. Ela define a estimativa de probabilidade com esta fórmula:

$$\widehat{P}(c_i|c_{i-2:i-1}) = \lambda_3 P(c_i|c_{i-2:i-1}) + \lambda_2 P(c_i|c_{i-1}) + \lambda_1 P(c_i)$$

onde os coeficientes antes das probabilidades devem somar 1. Os valores dos parâmetros podem ser fixados ou treinados com um algoritmo de maximização de expectativa (EM). Também é possível ter os valores de parâmetros dependentes do resultado: se tivermos um resultado de trigrama alto, atribuímos a ele um peso relativamente maior; se for apenas um resultado baixo, colocamos mais peso nos modelos de bigrama e unigrama. Um grupo de pesquisadores desenvolveu modelos de alisamento cada vez mais sofisticados, enquanto o outro grupo sugeriu reunir um corpus maior de modo que mesmo os modelos de alisamento simples funcionassem bem. Ambos estão tentando atingir a mesma meta: reduzir a variância no modelo de linguagem.

Em resumo, tudo não passa de Matemática e Estatística.

Com tantos modelos possíveis de n-grama, bigrama, trigrama, com interpolação de alisamento com valores diferentes de λ etc., como saber que modelo escolher? Podemos avaliar um modelo com validação cruzada. Dividir o corpus em um corpus de treinamento e um corpus de validação. Determinar o modelo de treinamento a partir dos dados de treinamento. Em seguida, avaliar o modelo pela validação do corpus. Ou seja, nada muito diferente do que fazemos em Machine Learning.

A avaliação pode ser uma métrica de tarefa específica, tal como medir a precisão da identificação da linguagem. Alternativamente, podemos ter um modelo de tarefas independentes de qualidade da linguagem: calcular a probabilidade atribuída ao corpus de validação pelo modelo; quanto maior for a probabilidade, melhor. Essa métrica é inconveniente porque a probabilidade de um grande corpus será um número muito pequeno, e um subfluxo de ponto flutuante torna-se um problema. Uma forma diferente de descrever a probabilidade de uma sequência é com uma medida chamada perplexidade, definida por esta fórmula:

$$Perplexidade\,(c_{1:N}) = P(c_{1:N})^{-\frac{1}{N}}$$

A perplexidade pode ser imaginada como o inverso da probabilidade, normalizada pelo comprimento da sequência. Pode também ser imaginada como o fator de ramificação médio ponderado de um modelo. Suponha que haja 100 caracteres em nossa língua, e nosso modelo informa que todos eles têm a mesma probabilidade. Em seguida, para uma sequência de qualquer tamanho, a perplexidade será 100. Se alguns caracteres forem mais prováveis do que outros, e o modelo reflete isso, o modelo terá uma perplexidade menor que 100. Esta pode ser uma fórmula de avaliação do modelo e falaremos mais sobre isso adiante.