kNNSumm: Um Sumarizador Automático de Documentos Utilizando Aprendizado Baseado em Instâncias

Carlos N. Silla Jr.[*]**, Celso A. A. Kaestner**

Pontifícia Universidade Católica do Paraná (PUC-PR)
Rua Imaculada Conceição 1155 – CEP 80215-901
Curitiba - Paraná - Brasil

{silla,kaestner}@ppgia.pucpr.br

Resumo. Neste trabalho é apresentada a arquitetura do kNNSumm (*k-NN Summarizer*), um sumarizador automático de documentos que utiliza o aprendizado de máquina baseado em instâncias. Também são apresentados os resultados obtidos com sua aplicação em uma coleção de documentos em inglês, extraídos da base TIPSTER, que é amplamente utilizada na literatura da área. Além disso, apresenta-se por meio de um exemplo simples e didático o funcionamento detalhado do sumarizador, e de uma forma geral também a tarefa de sumarização quando tratada por uma abordagem de aprendizado de máquina.

Abstract. In this work is presented the architecture of kNNSumm (*k-NN Summarizer*), an automatic document summarizer based on a instance based machine learning approach. The results achieved by its use on a document collection of english documents extracted from the TIPSTER base which is widely used in the literature are presented also. Additionally, we present a simple and didactic example of the procedures used by the summarizer, and in a more general way the text summarization task with machine learning.

Palavras-chave: Processamento Automático de Documentos, Sumarização Automática de Documentos, Sumarização de documentos com Aprendizado de Máquina, Sumarização com Aprendizado Baseado em Instâncias.

Keywords: Automatic Text Processing, Automatic Text Summarization, Machine Learning Text Summarization, Instace Based Summarization.

Submissão ao "V Workshop de Agentes y Sistemas Inteligentes (WASI)"

[*]Bolsista PIBIC - CNPq

1. Introdução

Definitivamente esta é a era da explosão da informação (*Information Explosion*). Um estudo recente de Berkeley [6] mostra que em 2002 havia 5 milhões de terabytes de novas informações criadas em documentos impressos, filmes, mídias ópticas e magnéticas. A Web sozinha contem cerca de 170 terabytes de informação, o que é equivalente a 17 vezes a coleção da biblioteca do congresso americano.

Porém é muito difícil fazer uso de toda essa informação. Muitos problemas tais como a busca, a recuperação e extração de informações e a sumarização automática de documentos se tornaram importantes tópicos de pesquisa na área de Ciência da Computação. Desta forma o uso de ferramentas automatizadas para o tratamento da informação passou a ser vital para o usuário, pois sem o auxílios destas ferramentas é virtualmente impossível explorar toda a informação relevante disponível na Web [18].

Nesse contexto, a tarefa de sumarização automática de documentos é muito importante. O objetivo de um sumarizador é gerar um sumário do texto original que permita ao usuário ter acesso as principais informações contidas naquele texto, mas num tempo de leitura muito inferior [7], visto que os sumários geralmente têm cerca de 10% a 30% do tamanho do texto original [8]. De uma forma geral os sumários são produzidos a partir de um conjunto de características que são obtidas empiricamente, por meio de técnicas estatísticas e/ou de processos oriundos do processamento de linguagem natural.

Uma das abordagens recentes para o problema de sumarização automática de documentos faz uso de algoritmos de aprendizado de máquina [4]. Nesse contexto a tarefa de sumarização é vista como um problema de classificação (aprendizado supervisionado), onde uma sentença do documento pode pertencer a 2 classes: *Pertencente* ou *Não Pertentence* ao sumário. Em trabalhos anteriores foram utilizados os classificadores Naïve Bayes [10] e C4.5 [12].

O objetivo deste trabalho é investigar a aplicação do aprendizado baseado em instâncias, utilizando o classificador k-NN, à tarefa de sumarização. Também se deseja verificar se este algoritmo é o mais adequado a tarefa do que os outros classificadores utilizados anteriormente, pois, como visto em [3] o uso do classificador para a tarefa de sumarização pode ser considerado tão importante quanto o conjunto de características que é utilizado.

Este trabalho também tem como objetivo apresentar, por meio de um exemplo simples e didático, o procedimento seguido em cada etapa do processo, de forma a permitir a compreensão do processo de sumarização automática que se fundamenta em aprendizado de máquina.

Este artigo está organizado da seguinte forma: na seção é descrito o procedimento utilizado pelo kNNSumm, para lidar com a tarefa de sumarização com um enfoque de classificação; a seção 3 descreve a tarefa de sumarização e as características utilizadas; a seção 4 apresenta os resultados computacionais obtidos; e na seção 5 discutem-se as principais conclusões do trabalho e indicam-se trabalhos futuros.

2. Sumarização com um enfoque de classificação

O *kNNSumm* surgiu da necessidade de estender as funcionalidades do *ClassSumm* (*Classification-based Summarizer*), que foi inicialmente proposto por Larocca Neto et al. [3], [4], e é um sistema para sumarização automática de documentos que utiliza procedimentos de aprendizagem de máquina e trata a sumarização de textos como uma tarefa de classificação. Diferentemente do *kNNSumm* o *ClassSumm* emprega outros mecanismos para a classificação.

Para utilizar um enfoque de aprendizado de máquina para a tarefa de sumarização, o sistema *kNNSumm* realiza as seguintes etapas durante o treinamento:

1. Identificação das sentenças do documento original;
2. Associação à cada sentença com um vetor de características previsoras, cujo os valores são obtidos diretamente a partir do conteúdo da própria sentença.
3. Associação à cada sentença com uma das seguintes classes: *Pertencente* ao sumário ou *Não Pertencente* ao sumário.

Esse procedimento permite realizar a tarefa de sumarização como se fosse uma tarefa de classificação. Como é comum na tarefa de classificação, o objetivo do algoritmo é descobrir, a partir dos dados, qual o relacionamento que prevê corretamente o valor de cada classe baseado nos valores das características previsoras daquela sentença. Mais precisamente, essa abordagem leva aos seguintes passos para resolver o problema:

1. O sistema constrói um conjunto de treinamento (a partir de uma coleção de documentos com seus respectivos sumários) onde cada exemplo corresponde a uma sentença do documento original, e cada exemplo é representado por um conjunto de características e uma classe determinada. O conjunto de treinamento é armazenado utilizando o padrão .arff [17].
2. Dado um novo conjunto de documentos, o sistema produz um conjunto de teste com características no mesmo formato que as do conjunto de treinamento, mas com os valores das classes desconhecidos.
3. Cada sentença no conjunto de testes é classificada, pelo algoritmo de classificação, em uma das duas classes: *Pertencente* ou *Não Pertencente* ao sumário.

3. O processo de sumarização

Nesta seção são descritas as etapas e os procedimentos normalmente utilizados em um sumarizador que emprega ferramentas de aprendizado de máquina. No intuito de facilitar a compreensão do processo, são apresentados os resultados obtidos em cada etapa com base no texto exemplo indicado à figura 1.

A tarefa de sumarização é composta de três etapas principais [16]:

1. Análise do Documento: nesta etapa é criada uma representação estruturada do documento. No caso do *kNNSumm* esta etapa é equivalente ao pré-processamento do documento, e a subsequente obtenção das características que são utilizadas no sistema;
2. Transformação do Documento: essa etapa também é chamada de Refinamento, onde, no caso do *kNNSumm*, se emprega o algoritmo de aprendizado baseado em instâncias.
3. Síntese do Documento: É nessa etapa que o sumarizador efetivamente gera o Sumário, a partir da escolha das sentenças mais adequadas para sua composição.

3.1. Análise do Documento

Durante a etapa de pré-processamento são realizados os seguintes procedimentos:

- O documento é dividido em sentenças, utilizando um dos procedimentos analisados em [15], visto que a correta identificação das sentenças é vital para se obter bons resultados na tarefa de sumarização [14].
- É efetuado o *case folding*, ou seja, todas as palavras são convertidas para a mesmo caso: maiúsculo ou minúsculo; no *kNNSumm* todos os caracteres do texto são convertidos para minúsculos, por exemplo “Charlie” vai ser convertido para “charlie”.

- São removidas as *stop words* do texto; *stop words* são palavras que ocorrem com muita freqüência no texto e por isso não carregam nenhuma informação semântica relevante; no *kNNSumm* as *stop words* que são removidas constam de uma lista de 524 palavras obtidas do código fonte da biblioteca BOW, desenvolvida na Universidade Carnegie Mellon.
- É efetuado o processo de obtenção do radical das palavras *(stemming)*, de forma a unificar elementos de texto com semântica similar; no *kNNSumm* utiliza-se o algoritmo de Porter [11], que converte, por exemplo, as palavras: "animals" e "animal" em "anim" , que é o radical das duas palavras.

O texto pré-processado está portanto dividido em suas sentenças constituintes, e são indicados apenas os radicais das palavras presentes no texto, ou *termos* do documento. Em seguida, obtém-se a partir deste texto pré-processado a representação vetorial do documento, forma de representação que é amplamente utilizada na área de recuperação de informações [13]. Um documento é representado por uma matriz onde cada linha corresponde a uma sentença e cada coluna corresponde a um termo do documento, e o valor de cada elemento da matriz é obtido a partir de uma função da freqüência do termo na sentença e no documento. Neste trabalho é utilizada a medida TF (*Term Frequency*), que corresponde simplesmente à freqüência em que o termo ocorre na sentença. Dessa forma, a repesentação vetorial do documento exemplo seria a indicada na figura 2.

A partir desta representação e do documento original com as sentenças identificadas podem ser extraídas características de dois tipos: (a) superficiais (*shallow*), que são obtidas por meio de cálculos estatísticos efetuados sobre os elementos do texto; e (b) profundas (*deep*), que utilizam alguma forma de conhecimento lingüístico na sua obtenção.

No *kNNSumm* as características superficiais são obtidas a partir da representação vetorial do documento:

- Posição relativa da sentença (pos): é a posição da sentença no texto, tendo um valor final normalizado entre 0 e 1. Considerando o texto de exemplo estes valores seriam: $pos(\mathrm{S}_1) = 0.25$; $pos(\mathrm{S}_2) = 0.5$; $pos(\mathrm{S}_3) = 0.75$; $pos(\mathrm{S}_4) = 1$.
- Tamanho da sentença ($size$): O tamanho da sentença é obtido contando-se o número de palavras em cada sentença; o valor final dessa característica é normalizado entre 0 e 1. Dessa forma: $size(\mathrm{S}_1) = 1$; $size(\mathrm{S}_2) = 0.5$; $size(\mathrm{S}_3) = 0.5$; $size(\mathrm{S}_4) = 1$.
- TF-ISF Médio da Sentença ($tsisf$): o TF-ISF (*Term-Frequency Inverse-Sentence-Frequency*) [5] é uma medida correspondente à clássica medida TF-IDF largamente empregada em Recuperação de Informações [13]. No caso do *kNNSumm* o TS-ISF de um termo w na sentença s é dado por: TF(w,s) * ISF(w,s), onde O TF(w,s) é a frequência do termo w na sentença s e o ISF(w,s) é dado por: log($|S|/SF(w)$) onde $|S|$ é o numero de sentenças do documento e SF(w) é o número de sentenças em que ocorre o termo w em todo o documento. Em seguida se calcula o TF-ISF médio de todos os termos presentes na sentença e normaliza-se o valor

```
            Título:  Animals in My House
              Palavra-chave:  animals
In my house there are many animals.
The one I like the most is my dog.
It's name is Charlie.
Charlie is the best dog ever.
```

Figura 1: Texto de Exemplo

	house	anim	dog	charli
T	1	1	0	0
K	0	1	0	0
S_1	1	1	0	0
S_2	0	0	1	0
S_3	0	0	0	1
S_4	0	0	1	1

Figura 2: Representação Vetorial do Texto de Exemplo

correspondente. No caso do exemplo tem-se: $tfisf(S_1) = 1$; $tfisf(S_2) = 0.5$; $tfisf(S_3) = 0.5$; $tfisf(S_4) = 0.5$.

- Similaridade com o Título ($simt$): para o cálculo desta característica utiliza-se a distância de similaridade do cosseno [13], que também é amplamente utilizada na área de Recuperação de Informações. Para uma sentença s e o título t a distância do cosseno será: $cos(s,t) = \frac{s.t}{|s||t|}$ onde as sentenças são encaradas como vetores e utilizam-se na forma habitual o produto escalar e o módulo dos vetores correspondentes. Considerando-se o exemplo, isto poderia ser reescrito da seguinte forma:

$$cos(S_1, T) = \frac{1*1+1*1+0*0+0*0}{\sqrt{(1^2+1^2+0^2+0^2)} * \sqrt{(1^2+1^2+0^2+0^2)}}$$

  Dessa forma, no exemplo: $simt(S_1) = 1$; $simt(S_2) = 0$; $simt(S_3) = 0$; $simt(S_4) = 0$.
- Similaridade com as Palavras-Chave ($simk$): esta característica é obtida de forma análoga à similaridade com o título; a única diferença é que se utiliza ao invés do título, a representação vetorial das palavras chaves (K) no cálculo das distâncias. Dessa forma: $simk(S_1) = 0.709$; $simk(S_2) = 0$; $simk(S_3) = 0$; $simk(S_4) = 0$.
- Semelhança entre as sentenças do documento ($sentsim$): para cada sentença é calculada a sua similaridade para com as demais sentenças do documento, por meio da distância do cosseno. A distância da sentença s e todas as demais é o valor "puro" dessa característica, que é em seguida normalizado entre 0 e 1. As sentenças com o valor de $sentsim$ próximo de 1 possuem um alto grau de coesão. Dessa forma, no exemplo: $sentsim(S_1) = 0$; $sentsim(S_2) = 0.5$; $sentsim(S_3) = 0.5$; $sentsim(S_4) = 1$.
- Semelhança entre as sentenças e o centróide do documento ($centroidsim$): o primeiro passo desse procedimento é obter o centróide do documento, que nada mais é do que o vetor médio de todas as sentenças do documento. Considerando o texto exemplo o centróide seria representado pelo vetor: (0.25; 0.25; 0.5; 0.5). Em seguida calcula-se o valor base da distância entre a sentença e o centróide do documento, utilizando a distância do cosseno, e finalmente os valores são normalizados entre 0 e 1. Sentenças com um valor próximo de 1 indicam um maior grau de coesão com o centróide do documento. Dessa forma, no exemplo: $centroidsim(S_1) = 0.5$; $centroidsim(S_2) = 0.709$; $centroidsim(S_3) = 0.709$; $centroidsim(S_4) = 1$.

Além destas características, a representação vetorial do documento também é utilizada para criar uma estrutura de dados que procura indicar de forma aproximada a árvore retórica do texto. Essa estrutura é obtida utilizando um algoritmo de agrupamento hierárquico (*Hierachical Clustering*), cuja saída é uma árvore binária onde os nós folhas representam as sentenças e os nós intermediários representam o relacionamento entre as sentenças. A medida de similaridade utilizada pelo algoritmo é, novamente, a medida do cosseno. A partir da árvore gerada são obtidas as seguintes características:

- Altura da Sentença na Árvore ($height$): é a altura de cada sentença s na árvore gerada pelo algoritmo de agrupamento hierárquico. Esse valor é normalizado entre 0 e 1, produzindo, no exemplo: $height(\mathrm{S}_1) = 0.5$; $height(\mathrm{S}_2) = 0.75$; $height(\mathrm{S}_3) = 1$; $height(\mathrm{S}_4) = 1$.

As próximas quatro características consideram a direção tomada pelo caminho que vai da raiz da árvore à sentença dada, em cada altura específica ($direction_i$). Os valores possíveis para esta característica são: *Left* caso o nó se encontre a esquerda na altura especificada, *Right* caso o nó se encontre à direita na altura especificada e *None* caso o nó não se encontre abaixo da altura especificada.

Para esta característica são consideradas as alturas 1, 2, 3 e 4, a partir da raiz da árvore. Dessa forma os valores em questão para o texto exemplo seriam:

- Altura 1: $direction_1(\mathrm{S}_1) =$ *Left*; $direction_1(\mathrm{S}_2) =$ *Right*; $direction_1(\mathrm{S}_3) =$ *Right*; $direction_1(\mathrm{S}_4) =$ *Right*.
- Altura 2: $direction_2(\mathrm{S}_1) =$ *None*; $direction_2(\mathrm{S}_2) =$ *Left*; $direction_2(\mathrm{S}_3) =$ *Right*; $direction_2(\mathrm{S}_4) =$ *Right*.
- Altura 3: $direction_3(\mathrm{S}_1) =$ *None*; $direction_3(\mathrm{S}_2) =$ *None*; $direction_3(\mathrm{S}_3) =$ *Left*; $direction_3(\mathrm{S}_4) =$ *Right*.
- Altura 4: $direction_4(\mathrm{S}_1) =$ *None*; $direction_4(\mathrm{S}_2) =$ *None*; $direction_4(\mathrm{S}_3) =$ *None*; $direction_4(\mathrm{S}_4) =$ *None*.

No contexto do *kNNSumm* as direções na árvore são mapeadas da seguinte forma: *None* é classificado como 0; *Left* é classificada como 0.5; e *Right* é classificada como 1.

Além das características superficiais, o kNNSumm também utiliza as seguintes características profundas, obtidas quando necessário diretamente do documento original com as sentenças identificadas:

- Indicador de conceitos principais ($mainconcepts$): essa característica binária indica se uma sentença possui ou não um dos indicadores de conceitos principais do texto. Esses conceitos são obtidos postulando-se que a maioria das idéias principais podem ser expressas por substantivos. Dessa forma, para cada sentença, são identificados todos os substantivos, utilizando um rotulador morfo-sintático (*Part-of-speech Tagger*) [1]; para cada substantivo calcula-se o número de sentenças em que o mesmo ocorre; os quinze substantivos mais freqüentes são selecionados como os indicadores de conceitos principais do documento. Dessa forma, esta característica vai ser considerada como sendo verdadeira para uma sentença se esta contiver pelo menos um desses quinze substantivos, e falsa caso contrário. No exemplo esses valores seriam: $mainconcepts(\mathrm{S}_1) =$ Verdadeiro; $mainconcepts(\mathrm{S}_2) =$ Verdadeiro; $mainconcepts(\mathrm{S}_3) =$ Verdadeiro; $mainconcepts(\mathrm{S}_4) =$ Verdadeiro.
- Presença de Nomes Próprios ($propernames$): essa característica indica se a sentença possui (verdadeiro) ou não (falso) um nome próprio, cuja indicação é obtida utilizando o rotulador morfo-sintático. Considerando o exemplo esses valores seriam: $propernames(\mathrm{S}_1) =$ Falso; $propernames(\mathrm{S}_2) =$ Falso; $propernames(\mathrm{S}_3) =$ Verdadeiro; $propernames(\mathrm{S}_4) =$ Verdadeiro.
- Presença de Anáforas ($anaphors$): essa característica binária indica se existe uma anáfora entre as seis primeiras palavras da sentença. As anáforas são obtidas a partir de uma lista fixa. Dessa forma, para o exemplo: $anaphors(\mathrm{S}_1) =$ Falso; $anaphors(\mathrm{S}_2) =$ Falso; $anaphors(\mathrm{S}_3) =$ Verdadeiro; $anaphors(\mathrm{S}_4) =$ Falso.
- Presença de Marcadores de Discurso ($speech$): essa característica binária é verdadeira caso a sentença possua uma marcador de discurso como “because” ou “futhermore” e falsa caso

contrário. No *kNNSumm* os marcadores de discurso constam de uma lista fixa. Dessa forma: $speech(\mathrm{S}_1) =$ Falso; $speech(\mathrm{S}_2) =$ Verdadeiro; $speech(\mathrm{S}_3) =$ Falso; $speech(\mathrm{S}_4) =$ Falso.

3.2. Transformação do Documento

Como mencionado anteriormente, o kNNSumm faz uso do algoritmo de classificação k-NN. Uma descrição detalhada do algoritmo k-NN pode ser encontrada em [2].

Como em muitos algoritmos de aprendizado de máquina para aplicar o k-NN é necessário se ter um conjunto de treinamento, isto é, um conjunto de instâncias que contenham além de suas características o indicativo da classe à qual pertencem. No contexto do kNNSumm é utilizado um arquivo .arff com as informações de treinamento onde cada uma das instâncias é uma sentença, e o vetor de características que representa essa sentença é composto das características apresentadas na subseção 3.1 acrescida da classe a qual pertencem, neste caso *Pertencente* ou *Não Pertencente* ao sumário.

Dessa forma, dado um conjunto de treinamento, para se lidar com novas instâncias é computada a distância entre esta nova instância e todas as instâncias disponíveis no conjunto de treinamento, utilizando-se, por exemplo, a distância euclidiana (1).

$$Dist(d_1, d_2) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{1}$$

A classificação de cada nova instância vai depender do valor do paramêtro k, pois para k = 1 cada nova instância vai ser classificada como sendo da classe correspondente àquela da instância mais próxima no conjunto de treinamento.

Para outros valores de k consideram-se as k instâncias mais próximas no conjunto de treinamento, que contribuem para a decisão da classe a atribuir; desta forma o ideal é se utilizar para k um valor ímpar, permitindo que a decisão seja feita por votação simples e evitando empates.

3.3. Síntese do Documento

Uma observação importante é que o procedimento descrito na seção anterior não leva em consideração o tamanho do sumário a ser gerado.

Na prática o usuário geralmente deseja obter um sumário de um tamanho determinado, ou seja, com uma taxa de compressão em relação ao documento original. Por isso, é importante utilizar um algoritmo que ao invés de simplesmente predizer a classe e gerar o sumário, consiga ordenar as sentenças de forma que possam ser escolhidas as sentenças mais relevantes para uma determinada taxa de compressão para compor o sumário.

Por esse motivo para ser possível ordenar as sentenças utilizando o classificador k-NN, utilizou-se o seguinte procedimento:

1. a cada sentença associou-se um valor obtido a partir da média harmônica das distâncias dos k vizinhos mais próximos.
2. as sentenças são ordenadas de acordo com estas distâncias, considerando a ordem crescente da classe *Pertencente* e em seguida as sentenças da classe *Não Pertencente* em ordem decrescente.

4. Resultados Computacionais

Os experimentos foram realizados utilizando uma base de documentos formadas por noticias extraídas do *Wall Street Journal* da coleção TIPSTER[9]. Essa coleção é comumente utilizada para testes na literatura de sumarização automática de documentos.

Para cada documento, o sumário considerado ideal, que permitiu a constituição do conjunto de treinamento, foi produzido utilizando-se um dos seguintes procedimentos: (1) o sumário ideal foi gerado automaticamente, e é formado pelas sentenças mais similares, de acordo com a medida do cosseno, com o sumário provido pelo autor do texto, seguindo o procedimento definido por Mani e Bloedorn [8]; esse tipo de sumário é referido como "sumário ideal automático"; (2) o sumário ideal foi gerado manualmente, produzido por uma professora de inglês que selecionou as sentenças mais relevantes do texto; esse tipo de sumário é aqui referido como "sumário ideal manual".

Em todos os experimentos o conjunto de treinamento foi composto de 100 documentos com seus respectivos sumários ideais automáticos. Foram realizados experimentos com dois tipos de bases de teste. A primeira contendo 100 documentos com seus respectivos sumários ideais automáticos, e a segunda contendo 30 documentos com seus respectivos sumários ideais manuais.

Em todos os experimentos o objetivo foi o de mensurar o acerto preditivo, ou seja, a habilidade de generalização do sumarizador. Como usual, os conjuntos de teste possuem apenas documentos não incluídos na etapa de treinamento.

Os resultados obtidos utilizando o classificador *kNNSumm* são apresentados nas Tabelas 1 e 2. Nelas também são apresentados os resultados obtidos em experimentos anteriores com o uso do *ClassSumm* [3], [4]. A Correção é a média obtida pelos valores de Precisão (*Precision*) e Cobertura (*Recall*) médios para cada conjunto de teste. A precisão é o quociente entre o número de sentenças selecionadas corretamente para compor o sumário e o número de sentenças selecionadas para compor o sumário. A cobertura é o quociente entre o número de sentenças selecionadas corretamente para compor o sumário e o número de sentenças do sumário ideal. Como nesses experimentos a taxa de compressão define um tamanho padrão para que os sumários gerados tenham o mesmo número de sentenças que os sumários alvos, precisão = cobertura.

Tabela 1: Correção para os documentos com sumários ideais automáticos

Método	10% Compressão	20% Compressão
Knn (k = 1)	17.43%	23.24%
Knn (k = 3)	18.87%	23.39%
Knn (k = 5)	19.97%	23.56%
Naïve Bayes	40.47%	51.43%
C4.5	22.36%	34.68%

Tabela 2: Correção para os documentos com sumários ideais manuais

Método	10% Compressão	20% Compressão
Knn (k = 1)	12.35%	18.92%
Knn (k = 3)	12.00%	19.01%
Knn (k = 5)	14.25%	17.34%
Naïve Bayes	26.14%	37.50%
C4.5	24.38%	31.73%

Os resultados apresentados nas tabelas 1 e 2 mostram que a correção obtida pelo classificador *kNNSumm* para a tarefa de sumarização foi muito aquém do esperado, tanto nos experimentos com a base de documentos com sumários ideais automáticos, quanto com a base dos sumários ideais manuais.

Esta conclusão reforça a observação de [3], que indica que a escolha do classificador é, para a tarefa de sumarização, tão importante quanto a escolha das características utilizadas.

5. Conclusões e Direções Futuras

Como mencionado anteriormente, o objetivo deste trabalho era verificar a eficiência do uso de algoritmo de aprendizado de máquina que utilizasse o aprendizado baseado em instâncias. Para isso foi utilizado o *kNNSumm*.

Os resultados obtidos mostraram que a performance do classificador k-NN foi muito aquém do esperado, visto que em todos os experimentos ele foi inferior aos classificadores bayesiano e C4.5, desenvolvidos em trabalhos anteriores. Esses resultados corroboram com as conclusões apresentadas no trabalho de [3], que indica ser a escolha do classificador para a tarefa de sumarização tão importante quanto a escolha das características utilizadas.

Como trabalho futuro os autores deverão aplicar outros métodos à tarefa, visando aumentar a taxa de acerto dos classificadores, tais como o uso de técnicas de *bagging* e *boosting*, além de outros métodos oriundos da área de Reconhecimento de Padrões.

Referências

[1] BRILL, E. A simple rule-based part-of-speech tagger. In *Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing* (Trento, IT, 1992), pp. 152–155.

[2] DUDA, R. O., HART, P. E., AND STORK, D. G. *Pattern Classification*. Wiley-Interscience, 2001.

[3] LAROCCA NETO, J. Contribuição ao estudo de técnicas para sumarização automática de textos. Master's thesis, Pontifícia Universidade Católica do Paraná (PUC-PR), Programa de Pós-Graduação em Informática Aplicada, 2002.

[4] LAROCCA NETO, J., FREITAS, A. A., AND KAESTNER, C. A. A. Automatic text summarization using a machine learning approach. In *XVI Brazilian Symp. on Artificial Intelligence* (2002), no. 2057 in Lecture Notes in Artificial Intelligence, pp. 205–215.

[5] LAROCCA NETO, J., SANTOS, A. D., KAESTNER, C. A. A., AND FREITAS, A. A. Document clustering and text summarization. In *Proc. 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining* (2000), pp. 41–55.

[6] LYMAN, P., AND VARIAN, H. R. How much information. Retrieved from http://www.sims.berkeley.edu/how-much-info-2003 on [01/19/2004], 2003.

[7] MANI, I. *Automatic Summarization*. John Benjamins Publishing Company, 2001.

[8] MANI, I., AND BLOEDORN, E. Machine learning of generic and user-focused summarization. In *Proc. of the 15th National Conf. on Artificial Intelligence (AAAI 98)* (1998), pp. 821–826.

[9] MANI, I., HOUSE, D., KLEIN, G., HIRSCHMAN, L., OBRSL, L., FIRMIN, T., CHRZANOWSKI, M., AND SUNDHEIM, B. The tipster summac text summarization evaluation. MITRE Technical Report MTR 98W0000138, The MITRE Corporation, October 1998.

[10] MITCHELL, T. M. *Machine Learning*. McGraw-Hill, 1997.

[11] PORTER, M. F. An algorithm for suffix stripping. *Program 14*, 3 (July 1980), 130–137.

[12] QUINLAN, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[13] SALTON, G., AND BUCKLEY, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management 24* (1988), 513–523.

[14] SILLA JR., C. N., AND KAESTNER, C. A. A. Estudo de métodos automáticos para sumarização de textos. In *Simpósio de Tecnologias de Documentos* (São Paulo, SP, Brazil, Setembro 2002), K. Weber and C. Kaestner, Eds., pp. 45–49.

[15] SILLA JR., C. N., AND KAESTNER, C. A. A. An analysis of sentence boundary detection systems for english and portuguese documents. In *5th International Conf. on Intelligent Text Processing and Computational Linguistics* (2004), no. 2945 in Lecture Notes in Computer Science, pp. 135–141.

[16] SPARCK-JONES, K. *Advances in Automatic Text Summarization*. MIT Press, 1999, ch. Automatic Summarizing: factors and directions, pp. 1–12.

[17] WITTEN, I. H., AND FRANK, E. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, 1999.

[18] ZHONG, N., LIU, J., AND YAO, Y. In search of the wisdom web. *IEEE Computer 35*, 11 (2002), 27–31.