feature is more general and can sometimes overcome errors propagated from the named-entity system.

To make this more concrete, the feature vectors for a document containing "John Smith" are highlighted in Figure 1. The superscript number for each phrase refers to the sentence where the phrase is located, and we assume that the syntactic phrase chunker and the EXERT system work perfectly.

3.3 Similarity matrix learning

Given a pair of feature vectors consisting of phrase-based features, we need to choose a similarity scheme to calculate the similarity. Because of the word-space delimiter in English, the feature vector for an English document comprises phrases, whereas that for a Chinese document comprises tokens. There are a number of similarity schemes for learning a similarity matrix from token-based feature vectors, but there are few schemes for phrase-based feature vectors.

Cohen et al. (2003) compared various similarity schemes for the task of matching English entity names and concluded that the hybrid scheme they call SoftTFIDF performs best. SoftTFIDF is a token-based similarity scheme that combines a standard TF-IDF weighting scheme with the Jaro-Winkler distance function. Since Chinese feature vectors are token-based, we can directly use SoftTFIDF to learn the similarity matrix. However, English feature vectors are phrase-based, so we need to run SoftTFIDF iteratively and call it "two-level SoftTFIDF." First, the standard SoftTFIDF is used to calculate the similarity between phrases in the pair of feature vectors; in the second phase, we reformulate the standard SoftTFIDF to calculate the similarity for the pair of feature vectors.

First, we introduce the standard SoftTFIDF. In a pair of feature vectors *S* and *T*, $S = (s_1, \dots, s_n)$ and $T = (t_1, \dots, t_m)$. Here, s_i (i = 1…n) and t_j (j = 1…m) are substrings (tokens). Let $CLOSE(\theta; S;T)$ be the set of substrings $w \in S$ such that there is some $v \in T$ satisfying $dist(w; v) > \theta$. The Jaro-Winkler distance function (Winkler, 1999) is $dist(;)$. For $w \in CLOSE(\theta; S;T)$, let $D(w; T) = \max_{v \in T} dist(w;v)$. Then the standard SoftTFIDF is computed as

$$\text{SoftTFIDF}(S,T) = \sum_{w \in CLOSE(\theta;S;T)} V(w, S) \times V(w, T) \times D(w, T)$$

$$V'(w, S) = \log(\text{TF}_{w,S} + 1) \times \log(\text{IDF}_w)$$

$$V(w, S) = \frac{V\ (w, S)}{\sqrt{\sum_{w \in S} V(w, S)^2}}$$

where $\text{TF}_{w,S}$ is the frequency of substrings *w* in *S*, and IDF_w is the inverse of the fraction of documents in the corpus that contain *w*. In computing the similarity for the English phrase-based feature vectors, in the second step of "two-level SoftTFIDF," the substring *w* is a phrase and *dist* is the standard SoftTFIDF.

So far, we have developed several feature models and learned the corresponding similarity matrices, but clustering usually needs only one unique similarity matrix. Since a feature may have different effects for the disambiguation depending on the ambiguous personal name in consideration, to achieve the best disambiguation ability, each personal name may need its own weighting scheme to combine the given similarity matrices. However, learning that kind of weighting scheme is very difficult, so in this paper, we simply combine the similarity matrices, assigning equal weight to each one.

3.4 Clustering

Although clustering is a well-studied area, a remaining research problem is to determine the optimal parameter setting during clustering, such as the number of clusters or the stop-threshold, a problem that is important for real tasks and that is not at all trivial.

Since the focus of this paper is only on feature development, we simply employ a clustering method that can reflect the quality of the similarity matrix for clustering. Here, we choose agglomerative clustering with a single linkage. Since each personal name may need a different parameter setting, to test the importance of the parameter setting for clustering, we use two kinds of stop-thresholds for agglomerative clustering in our experiments: first, to find the optimal stop-threshold for any ambiguous personal name and for each feature model, we run agglomerative clustering with all possible stop-thresholds, and choose the one that has the best performance as the optimal