Proceedings of the 6th International Joint Conference on
Natural Language Processing (IJCNLP), Nagoya, Japan, October 2013, pp. 1366--1374.

Linguistically Aware Coreference Evaluation Metrics

Chen Chen and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{yzcchen,vince}@hlt.utdallas.edu

Abstract

Virtually all the commonly-used evaluation metrics for entity coreference resolution are linguistically agnostic, treating the mentions to be clustered as generic rather than linguistic objects. We argue that the performance of an entity coreference resolver cannot be accurately reflected when it is evaluated using linguistically agnostic metrics. Consequently, we propose a framework for incorporating linguistic awareness into commonly-used coreference evaluation metrics.

1 Introduction

Coreference resolution is the task of determining which mentions in a text or dialogue refer to the same real-world entity. Designing appropriate evaluation metrics for coreference resolution is an important and challenging task. Since there is no consensus on which existing coreference evaluation metric is the best, the organizers of the CoNLL-2011 and CoNLL-2012 shared tasks on unrestricted coreference (Pradhan et al., 2011, 2012) decided to take the average of the scores computed by three coreference evaluation metrics, MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and $CEAF_e$ (Luo, 2005), as the official score of a participating coreference resolver.

One weakness shared by virtually all existing coreference evaluation metrics is that they are *linguistically agnostic*, treating the mentions to be clustered as generic rather than linguistic objects. In other words, while MUC, B^3, and CEAF were designed for evaluating coreference resolvers, their linguistic agnosticity implies that they can be used to evaluate *any* clustering task, including those that are not linguistic in nature.[1]

To understand why linguistic agnosticity is a potential weakness of existing scoring metrics, consider a document in which there are three coreferent mentions, *Hillary Clinton*, *she*, and *she*, appearing in this order in the document. Assume that two coreference resolvers, R_1 and R_2, are applied to these three mentions, where R_1 only posits *Hillary Clinton* and *she* as coreferent, and R_2 only posits the two occurrences of *she* as coreferent. Being linguistically agnostic, existing scoring metrics will assign the *same* score to both resolvers after seeing that both of them correctly assign two of the three objects to the same cluster. Intuitively, however, R_1 should receive a higher score than R_2: R_1 has facilitated automated text understanding by successfully finding the referent of one of the pronouns, whereas from R_2's output we know nothing about the referent of the two pronouns. Failure to rank R_1 higher than R_2 implies that existing scoring metrics fail to adequately reflect the performance of a resolver.[2]

Our goal in this paper is to address the aforementioned weakness by proposing a framework for incorporating linguistic awareness into the most commonly-used coreference scoring metrics, including MUC, B^3, and CEAF. Rather than making different modifications to different metrics, one of the contributions of our work lies in the proposal of a *unified* framework that enables us to employ the *same* set of modifications to create linguistically aware versions of all these metrics.

2 Existing Evaluation Metrics

In this section, we review four scoring metrics, MUC, B^3, and the two versions of CEAF, namely,

[1]This statement is also true for BLANC (Recasens and Hovy, 2011), a Rand Index-based coreference evaluation metric we will not focus on in this paper.

[2]One may disagree that R_1 should be ranked higher than R_2 by arguing that successful identification of two coreferential pronouns is not necessarily easier than resolving an anaphoric pronoun to a non-pronominal antecedent. Our argument, however, is based on the view traditionally adopted in pronoun resolution research that resolving an anaphoric pronoun entails finding a non-pronominal antecedent for it.

CEAF$_m$ and CEAF$_e$. As F-score is always computed as the unweighted harmonic mean of recall and precision, we will only show how recall and precision are computed. Note that unlike previous discussion of these metrics, we present them in a way that reveals their common elements.

2.1 Notation and Terminology

In the rest of this paper, we use the terms *coreference chains* and *coreference clusters* interchangeably. For a coreference chain C, we define $|C|$ as the number of mentions in C. *Key chains* and *system chains* refer to gold coreference chains and system-generated coreference chains, respectively. In addition, $\mathcal{K}(d)$ and $\mathcal{S}(d)$ refer to the set of gold chains and the set of system-generated chains in document d, respectively. Specifically,

$$\mathcal{K}(d) = \{K_i : i = 1, 2, \cdots, |\mathcal{K}(d)|\},$$

$$\mathcal{S}(d) = \{S_j : j = 1, 2, \cdots, |\mathcal{S}(d)|\},$$

where K_i is a chain in $\mathcal{K}(d)$ and S_j is a chain in $\mathcal{S}(d)$. $|\mathcal{K}(d)|$ and $|\mathcal{S}(d)|$ are the number of chains in $\mathcal{K}(d)$ and $\mathcal{S}(d)$, respectively.

2.2 MUC (Vilain et al., 1995)

MUC is a link-based metric. Given a document d, recall is computed as the number of common links between the key chains and the system chains in d divided by the number of links in the key chains. Precision is computed as the number of common links divided by the number of links in the system chains. Below we show how to compute (1) the number of common links, (2) the number of key links, and (3) the number of system links.

To compute the number of common links, a partition $P(S_j)$ is created for each system chain S_j using the key chains. Specifically,

$$P(S_j) = \{C_j^i : i = 1, 2, \cdots, |\mathcal{K}(d)|\} \quad (1)$$

Each subset C_j^i in $P(S_j)$ is formed by intersecting S_j with K_i. Note that $|C_j^i| = 0$ if S_j and K_i have no mentions in common. Since there are $|\mathcal{K}(d)| * |\mathcal{S}(d)|$ subsets in total, the number of common links is

$$c(\mathcal{K}(d), \mathcal{S}(d)) = \sum_{j=1}^{|\mathcal{S}(d)|} \sum_{i=1}^{|\mathcal{K}(d)|} w_c(C_j^i), \quad (2)$$

where $w_c(C_j^i) = \begin{cases} 0 & \text{if } |C_j^i| = 0; \\ |C_j^i| - 1 & \text{if } |C_j^i| > 0. \end{cases}$

Intuitively, $w_c(C_j^i)$ can be interpreted as the "weight" of C_j^i. In MUC, the weight of a cluster is defined as the *minimum* number of *links* needed to create the cluster, so $w_c(C_j^i) = |C_j^i| - 1$ if $|C_j^i| > 0$.

The number of links in the key chains, $\mathcal{K}(d)$, is calculated as:

$$k(\mathcal{K}(d)) = \sum_{i=1}^{|\mathcal{K}(d)|} w_k(K_i), \quad (3)$$

where $w_k(K_i) = |K_i| - 1$. The number of links in the system chains, $s(\mathcal{S}(d))$, is calculated as:

$$s(\mathcal{S}(d)) = \sum_{j=1}^{|\mathcal{S}(d)|} w_s(S_j), \quad (4)$$

where $w_s(S_j) = |S_j| - 1$.

2.3 B^3 (Bagga and Baldwin, 1998)

One of MUC's shortcoming is that it fails to reward successful identification of singleton clusters. To address this weakness, *B*^{3} first computes the recall and precision for each mention, and then averages these per-mention values to obtain the overall recall and precision.

Let m_n be the nth mention in document d. Its recall, $R(m_n)$, and precision, $P(m_n)$, are computed as follows. Let K_i and S_j be the key chain and the system chain that contain m_n, respectively, and let C_j^i be the set of mentions appearing in both S_j and K_i.

$$R(m_n) = \frac{w_c(C_j^i)}{w_k(K_i)}, P(m_n) = \frac{w_c(C_j^i)}{w_s(S_j)}, \quad (5)$$

where $w_c(C_j^i) = |C_j^i|$, $w_k(K_i) = |K_i|$, and $w_s(S_j) = |S_j|$.

2.4 CEAF (Luo, 2005)

While B^3 addresses the shortcoming of MUC, Luo presents counter-intuitive results produced by B^3, which it attributes to the fact that B^3 may use a key/system chain more than once when computing recall and precision. To ensure that each key/system chain will be used at most once in the scoring process, his CEAF scoring metric scores a coreference partition by finding an optimal *one-to-one mapping* (or *alignment*) between the chains in $\mathcal{K}(d)$ and those in $\mathcal{S}(d)$.

Since the mapping is one-to-one, not all key chains and system chains will be involved in it. Let

$\mathcal{K}_{min}(d)$ and $\mathcal{S}_{min}(d)$ be the set of key chains and the set of system chains involved in the alignment, respectively. The alignment can be represented as a one-to-one mapping function g, where

$$g(K_i) = S_j, K_i \in \mathcal{K}_{min}(d) \text{ and } S_j \in \mathcal{S}_{min}(d).$$

The score of g, $\Phi(g)$, is defined as

$$\Phi(g) = \sum_{K_i \in \mathcal{K}_{min}(D)} \phi(K_i, g(K_i)),$$

where ϕ is a function that computes the *similarity* between a gold chain and a system chain. The optimal alignment, g^*, is the alignment whose Φ value is the largest among all possible alignments, and can be computed efficiently using the Kuhn-Munkres algorithm (Kuhn, 1955).

Given g^*, the recall (R) and precision (P) of a system partition can be computed as follows:

$$R = \frac{\Phi(g^*)}{\sum_{i=1}^{|\mathcal{K}(d))|} \phi(K_i, K_i)}, P = \frac{\Phi(g^*)}{\sum_{j=1}^{|\mathcal{S}(d))|} \phi(S_j, S_j)}.$$

As we can see, at the core of CEAF is the similarity function ϕ. Luo defines two different ϕ functions, ϕ_3 and ϕ_4:

$$\phi_3(K_i, S_j) = |K_i \cap S_j| = w_c(C_j^i) \quad (6)$$

$$\phi_4(K_i, S_j) = \frac{2|K_i \cap S_j|}{|K_i| + |S_j|} = \frac{2 * w_c(C_j^i)}{w_k(K_i) + w_s(S_j)} \quad (7)$$

ϕ_3 and ϕ_4 result in mention-based CEAF (a.k.a. CEAF$_m$) and entity-based CEAF (a.k.a. CEAF$_e$), respectively.

2.5 Common functions

Recall that the three weight functions, w_c, w_k, and w_s, are involved in all the scoring metrics we have discussed so far. To summarize:

- $w_c(C_j^i)$ is the weight of the common subset between K_i and S_j. For MUC, its value is 0 if C_j^i is empty and $|C_j^i| - 1$ otherwise; for B^3, CEAF$_m$ and CEAF$_e$, its value is $|C_j^i|$.
- $w_k(K_i)$ is the weight of key chain K_i. For MUC, its value is $|K_i| - 1$, while for B^3, CEAF$_m$ and CEAF$_e$, its value is $|K_i|$.
- $w_s(S_j)$ is the weight of system chain S_j. For MUC, its value is $|S_j| - 1$, while for B^3, CEAF$_m$ and CEAF$_e$, its value is $|S_j|$.

Next, we will show that simply by redefining these three functions appropriately, we can create linguistically aware versions of MUC, B^3, CEAF$_m$, and CEAF$_e$.[3] For convenience, we will refer to their linguistically aware counterparts as LMUC, LB^3, LCEAF$_m$, and LCEAF$_e$.[4]

3 Incorporating Linguistic Awareness

As mentioned in the introduction, one of the contributions of our work lies in identifying the three weight functions that are common to MUC, B^3, CEAF$_m$, and CEAF$_e$ (see Section 2.5). To see why these weight functions are important, note that *any interaction between a scoring metric and a coreference chain is mediated by one of these weight functions*. In other words, if these weight functions are linguistically agnostic (i.e., they treat the mentions as generic rather than linguistic objects when assigning weights), the scoring metric that employs them will be linguistically agnostic. On the other hand, if these weight functions are linguistically aware, the scoring metric that employs them will be linguistically aware.

This observation makes it possible for us to design a *unified* framework for incorporating linguistic awareness into existing coreference scoring metrics. Specifically, rather than making different modifications to different scoring metrics to incorporate linguistic awareness, we can simply incorporate linguistic awareness into these three weight functions. So when they are being used in different scoring metrics, we can handily obtain the linguistically aware versions of these metrics.

In the rest of this section, we will suggest one way of implementing linguistic awareness. This is by no means the only way to implement linguistic awareness, but we believe that this is a good starting point, which hopefully will initiate further discussions in the coreference community.

3.1 Formalizing Linguistic Awareness

Other than illustrating the notion of linguistic awareness via a simple example in the introduction, we have thus far been vague about what ex-

[3] Note that for a given scoring metric, $w_c(C) = w_k(C) = w_s(C)$ for any non-empty chain C. The reason why we define three weight functions as opposed to one is that they are defined differently in the linguistically aware scoring metrics, as we will see.

[4] Our implementation of the linguistically aware evaluation metrics is available from `http://www.hlt.utdallas.edu/~yzcchen/coreference`.

actly it is. In this section, we will make this notion more concrete.

Recall that the goal of (co)reference resolution is to facilitate automated text understanding by finding the referent for each referring expression in a text. Hence, when resolving a mention, a resolver should be rewarded more if the selected antecedent allows the underlying *entity* to be inferred than if it doesn't, because the former contributes more to understanding the corresponding text than the latter. Note that the more *informative* the selected antecedent is, the easier it will be for the reader to infer the underlying entity. Here, we adopt a simple notion of linguistic informativeness based on the mention type: a name is more informative than a nominal, which in turn is more informative than a pronoun.[5] Hence, a coreference link involving a name should be given a higher weight than one that doesn't, and a coreference link involving a nominal should be given a higher weight than one that involves only pronouns.

We implement this observation by assigning to each link e_l a weight of $w_l(e_l)$, where $w_l(e_l)$ is defined using the first rule applicable to e_l below:
Rule 1: If e_l involves a name, $w_l(e_l) = w_{nam}$.
Rule 2: If e_l involves a nominal, $w_l(e_l) = w_{nom}$.
Rule 3: $w_l(e_l) = w_{pro}$.

There is a caveat, however. By assigning weights to coreference *links* rather than mentions, we will be unable to reward successful identification of singleton clusters, since they contain *no* links (and hence they carry no weights). To address this problem, we introduce a singleton weight w_{sing}, which will be assigned to any chain that contains exactly one mention.

So far, we have introduced four weights, $W = (w_{nam}, w_{nom}, w_{pro}, w_{sing})$, which encode our (somewhat simplistic) notion of linguistic awareness. Below we show how these four weights are incorporated into the three weight functions, w_c, w_k, and and w_s, to create their linguistically aware counterparts, w_c^L, w_k^L, and w_s^L.

3.2 Defining w_c^L

Recall that C_j^i represents the set of mentions common to key chain K_i and system chain S_j. To define the linguistically aware weight function $w_c^L(C_j^i)$, there are three cases to consider:

Case 1: $|C_j^i| \geq 2$
Recall that the linguistically agnostic w_c function returns a weight of $|C_j^i| - 1$. This makes sense, because in a linguistically agnostic situation, all the links have the same weight, and hence the weight assigned to C_j^i will be the same regardless of which $|C_j^i| - 1$ links in C_j^i are chosen. However, the same is no longer true in a linguistically aware setting: since the links may not necessarily have the same weight, the weight assigned to C_j^i depends on which $|C_j^i| - 1$ links are chosen. In this case, it makes sense for our linguistically aware w_c^L function to find the $|C_j^i| - 1$ links that have the largest weights and assign to w_c^L the sum of these weights, since they reflect how well a resolver managed to find informative antecedents for the mentions. Note that the sum of the $|C_j^i| - 1$ links that have the largest weights is equal the weight of the maximum spanning tree defined over the mentions in C_j^i.

Case 2: $|C_j^i| = 0$
In this case C_j^i is empty, meaning that K_i and S_j do not have any mention in common. w_c^L simply returns a weight of 0 when applied to C_j^i.

Case 3: $|C_j^i| = 1$
In this case, K_i and S_j have one mention in common. The question, then, is: can we simply return w_{sing}, the weight associated with a singleton cluster? The answer is no: since w_{sing} was created to reward *successful* identification of singleton clusters, a resolver should be rewarded by w_{sing} only if it correctly identifies a singleton cluster. In other words, w_c^L returns w_{sing} if all of C_j^i, K_i and S_j contain exactly one mention (which implies that the singleton cluster C_j^i is correctly identified); otherwise, w_c^L returns 0.

The definition of w_c^L is summarized as follows, where E is the set of edges in the maximum spanning tree defined over the mentions in C_j^i.

$$w_c^L(C_j^i) = \begin{cases} \sum_{e_l \in E} w_l(e_l) & \text{if } |C_j^i| > 1; \\ w_{sing} & \text{if } |C_j^i|, |K_i|, |S_j| = 1; \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

3.3 Defining w_k^L

Recall that w_k^L aims to compute the weight of key chain K_i. Given the definition of w_c^L, in order to ensure that the maximum recall is 1, it is natural to define w_K^L as follows, where E is the set of edges

[5] Different notions of linguistic informativeness might be appropriate for different natural language applications. In our framework, a different notion of linguistic informativeness can be implemented simply by altering the weight functions.

appearing in the maximum spanning tree defined over the mentions in K_i.

$$w_k^L(K_i) = \begin{cases} \sum_{e_l \in E} w_l(e_l) & \text{if } |K_i| > 1; \\ w_{sing} & \text{if } |K_i| = 1. \end{cases} \quad (9)$$

3.4 Defining w_s^L

Finally, we define w_s^L, the function for computing the weight of system chain S_j. To better understand how we might want to define w_s^L, recall that in MUC, B^3, and both versions of CEAF, precision and recall play a symmetric role. In other words, precision is computed by reversing the roles of the key partition $\mathcal{K}(d)$ and the system partition $\mathcal{S}(d)$ used to compute recall for document d. If we wanted precision and recall to also play a symmetric role in the linguistically aware versions of these scoring metrics, it would be natural to define w_s^L in the same way as w_k^L, where E is the set of edges appearing in the maximum spanning tree defined over the mentions in S_j.

$$w_s^L(S_j) = \begin{cases} \sum_{e_l \in E} w_l(e_l) & \text{if } |S_j| > 1; \\ w_{sing} & \text{if } |S_j| = 1. \end{cases} \quad (10)$$

However, there is a reason why it is undesirable for us to define w_s^L in this manner. Consider the special case in which a system partition $\mathcal{S}(d)$ contains only correct links, some of which are suboptimal.[6] Although $\mathcal{S}(d)$ contains only correct links, the precision computed by any scoring metric that employs w_s^L with the above definition will be less than one simply because it contains suboptimal links. In other words, if a scoring metric employs w_s^L with the above definition, it will penalize a resolver for choosing suboptimal links twice, once in recall and once in precision.

To avoid penalizing a resolver for the same mistake twice, w_s^L cannot be defined in the same way as w_k^L.[7] In particular, only spurious links (i.e., links between two non-coreferent mentions), not suboptimal links, should be counted as precision errors. To avoid this problem, recall that $P(S_j)$ is defined as a partition of system chain S_j created by intersecting S_j with all key chains in $\mathcal{K}(d)$.

$$P(S_j) = \{C_j^i : i = 1, 2, \cdots, |\mathcal{K}(d)|\}$$

Note that a link is spurious if it links a mention in $C_j^{i_1}$ with a mention in $C_j^{i_2}$, where $1 \leq i_1 \neq i_2 \leq \mathcal{K}(d)$. Without loss of generality, assume that there are ne_j non-empty clusters in $P(S_j)$. Note that we need $ne_j - 1$ spurious links in order to connect the ne_j non-empty clusters. To adequately reflect the damage created by these spurious links, among the different sets of $ne_j - 1$ spurious links that connect the ne_j non-empty clusters in $P(S_j)$, we choose the set where the sum of the weights of the links is the largest and count the edges in it as precision errors. We denote this set as $E_t(S_j)$.

Now we are ready to define w_s^L. There are two cases to consider.

Case 1: $|S_j| > 1$

In this case, $w_s^L(S_j)$ is computed as follows:

$$w_s^L(S_j) = \sum_{C_j^i \in P(S_j)} w_c^L(C_j^i) + \sum_{e \in E_t(S_j)} w_l(e). \quad (11)$$

Note that the second term corresponds to the precision errors discussed in the previous paragraph, whereas the first term corresponds to the sum of the values returned by w_c^L when applied to each cluster in $P(S_j)$. The first term guarantees that a resolver is penalized for precision errors because of spurious links, not suboptimal links.

Case 2: $|S_j| = 1$

In this case, S_j only contains one mention. We set $w_s^L(S_j)$ to w_{sing}.

4 Evaluation

In this section, we design experiments to better understand our linguistically aware metrics (henceforth *LMetrics*). Specifically, our evaluation is driven by two questions. First, given that the *LMetrics* are parameterized by a vector of four weights W, how do their behaviors change as we alter W? Second, how do the *LMetrics* differ from the existing metrics (henceforth *OMetrics*)?

4.1 Experimental Setup

We use as our running example the paragraph shown in Figure 1, which is adapted from the Bible domain of the English portion of the OntoNotes v5.0 corpus. There are 19 mentions in the paragraph, each of which is enclosed in parentheses and annotated as m_x^y, where y is the ID of the chain to which this mention belongs, and x is the mention ID.

Figure 2 shows five system responses (a–e) for our running example along with the key chains.

[6] Suboptimal links are links that are correct but do not appear in a maximum spanning tree for any of its chains.

[7] This implies that precision and recall will no longer play a symmetric role in our linguistically aware scoring metrics.

(Jesus)^1_a came near (Jerusalem)^2_d. Looking at (the city)^2_e, (he)^1_b began to cry for (it)^2_f and said, (I)^1_c wish (you)^2_g knew what would bring (you)^2_h (peace)^4_p. But it is hidden from (you)^2_i (now)^5_q. (A time)^6_r is coming when ((your)^2_j enemies)^3_n will hold (you)^2_k in on (all sides)^7_s. (They)^3_o will destroy (you)^2_l and (all (your)^2_m people)^8_t.

Figure 1: A paragraph adapted from the Bible domain of the OntoNotes 5.0 corpus.

For conciseness, a mention is denoted by its mention ID, and each connected sub-graph forms one coreference chain. Moreover, the type of a mention is denoted by its shape: a *square* denotes a NAME mention; a *triangle* denotes a NOMINAL mention, and a *circle* denotes a PRONOUN mention. Note that S_{you}, the set of coreferent "you" mentions consisting of $\{m_g^2, m_h^2, \cdots, m_m^2\}$, appears in all system responses.

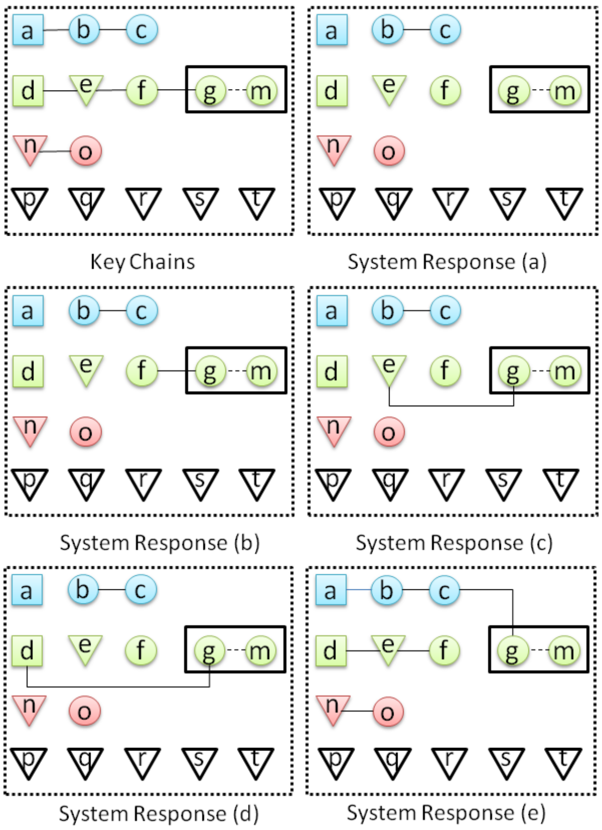

Figure 2: Key and system coreference chains.

Let us begin by describing the five system responses. Response (a) is produced by a simple and conservative resolver. Besides forming S_{you}, this resolver also correctly links m_b^1 with m_c^1. Responses (b), (c) and (d) each improves upon response (a) by linking S_{you} to one of three preceding mentions, namely, one PRONOUN mention, one NOMINAL mention, and one NAME mention respectively. Response (e) is produced by an aggressive resolver that tries to resolve all the pronouns to a non-pronominal antecedent, but unfortunately, it wrongly connects S_{you} to m_a^1, m_b^1 and m_c^1.

Next, we investigate the two questions posed at the beginning of Section 4.1. To determine how the *LMetrics* behave when used in combination with different weight vectors $W = (w_{nam}, w_{nom}, w_{pro}, w_{sing})$, we experiment with:

$W_1 = (1.0, 1.0, 1.0, 10^{-20})$;[8]
$W_2 = (1.0, 1.0, 1.0, 0.5)$;
$W_3 = (1.0, 1.0, 1.0, 1.0)$;
$W_4 = (1.0, 0.75, 0.5, 1.0)$;
$W_5 = (1.0, 0.5, 0.25, 1.0)$.

Note that W_1, W_2, and W_3 differ only with respect to w_{sing}, so comparing the results obtained using these weight vectors will reveal the impact of w_{sing} on the *LMetrics*. On the other hand, W_4 and W_5 differ with respect to the gap of the weights associated with the three types of mentions. Examining the *LMetrics* when they are used in combination with W_4 and W_5 will reveal the difference between having "relatively similar" weights versus having "relatively different" weights on the three mention types.

Figure 3 shows four graphs, one for each of the four *LMetrics*. Each graph contains six curves, five of which correspond to curves generated by using the aforementioned five weight vectors, and the remaining one corresponds to the *OMetric* curve that we include for comparison purposes. Each curve is plotted using five points that correspond to the five system responses.

4.2 Impact of w_{sing}

We first investigate the impact of w_{sing}. We will determine how the *LMetrics* behave in response to W_1, W_2 and W_3.

The first graph in Figure 3 shows the LMUC and MUC F-scores. As we can see, the scores of MUC and LMUC(W_1) are almost the same. This is understandable: the uniform edge weights and a very small w_{sing} in W_1 imply that LMUC will

[8] We set w_{sing} to a very small value other than 0, because setting w_{sing} to 0 may cause the denominator of the expressions in (5) and (7) to be 0.

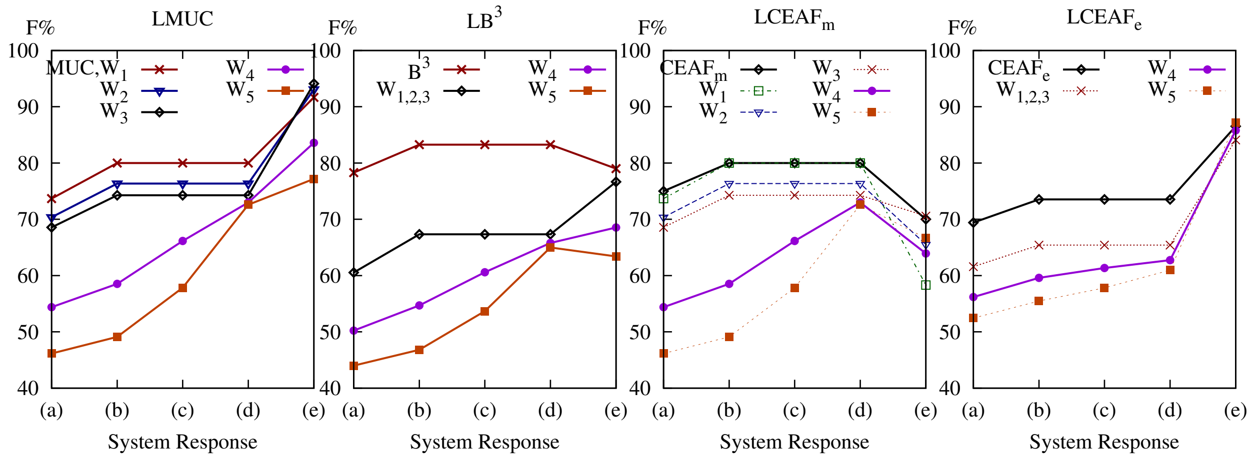

Figure 3: Comparison of the *LMetrics* scores under different weight settings and the *OMetrics* scores.

essentially ignore correct identification of single clusters and consider all errors to be equal, just like MUC. When we replace W_1 with W_2 and W_3, the two weight vectors with a larger w_{sing} value, and rescore the five responses, we see that the LMUC scores for responses (a), (b), (c) and (d) decrease. This is because LMUC uses w_{sing} to penalize these four responses for identifying wrong singleton clusters. On the other hand, the LMUC score for response (e) is higher than the corresponding MUC score, because LMUC additionally rewards response (e) for correctly classifying all singleton clusters without introducing erroneous singleton clusters.

The second graph in Figure 3 shows the LB^3 and B^3 F-scores. Here, we see that the scores for LB$^3(W_1)$, LB$^3(W_2)$ and LB$^3(W_3)$ are identical. These results suggest that the value of w_{sing} does not affect the LB^3 score, despite the fact that LB^3 does take into account singleton clusters when scoring, a property that it inherits from B^3. The reason is that regardless of what w_{sing} is, if a mention m is correctly classified as a singleton mention, both of $R(m)$ and $P(m)$ will be 1, otherwise, both will be 0 (see formula (5)). Note, however, that there is a difference between LB^3 and B^3: for an erroneously identified singleton cluster containing mention m, LB^3 sets $P(m)$ to 0 while B^3 sets $P(m)$ to 1. In other words, LB^3 puts a higher penalty on precision given erroneous singleton clusters. This difference causes LB^3 and B^3 to evaluate responses (a) and (e) differently. Recall that responses (a) and (e) are quite different: response (e) correctly finds informative antecedents for m_b^1, m_c^1, m_e^2, m_f^2 and m_o^3, whereas response (a) contains many erroneous singleton clusters. Despite the large differences in these responses, B^3 only gives 0.7% more points to response (e) than response (a). On the other hand, LB^3 assigns a much lower score to response (a) owing to the numerous erroneous singleton clusters it contains.

The third graph of Figure 3 shows the LCEAF$_m$ and CEAF$_m$ F-scores. Since LCEAF$_m$ uses both singleton and non-singleton clusters when computing the optimal alignment, it should not be surprising that as we increase w_{sing}, the singleton clusters will play a more important role in the LCEAF$_m$ score. Consider, for example, LCEAF$_m(W_1)$. Since $w_{sing} = 0$, LCEAF$_m(W_1)$ ignores the correct identification of singleton clusters. From the graph, we see that LCEAF$_m(W_1)$ gives a higher score to response (a) than response (e). This is understandable: response (a) is not penalized for the many erroneous singleton clusters it contains; on the other hand, response (e) is penalized for the erroneous coreference links it introduces. Now, consider LCEAF(W_3), where $w_{sing} = 1$. Here, response (e) is assigned a higher score by LCEAF(W_3) than response (a): response (a) is heavily penalized because of the many erroneous clusters it contains.

The rightmost graph of Figure 3 shows the LCEAF$_e$ and CEAF$_e$ F-scores. Like LB^3, LCEAF$_e$ returns the same score when it is used in combination with W_1, W_2 and W_3, because the ϕ_4 similarity function returns 0 or 1 when the key cluster or the system cluster it is applied to is a singleton cluster, regardless of the value of w_{sing}. In addition, we can see that LCEAF$_e$ penalizes erroneous singleton clusters more than CEAF$_e$ does

ch-ains	MUC			LMUC			B^3			LB^3			$CEAF_m$			$LCEAF_m$			$CEAF_e$			$LCEAF_e$		
	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F	R	P	F
(a)	58.3	100	73.7	50.7	58.6	54.4	64.3	100	78.3	39.2	70.0	50.2	75.0	75.0	75.0	50.7	58.6	54.4	91.1	56.1	69.4	73.8	45.4	56.2
(b)	66.7	100	80.0	53.7	64.3	58.5	71.3	100	83.3	43.1	75.0	54.7	80.0	80.0	80.0	53.7	64.3	58.5	91.9	61.3	73.6	74.5	49.7	59.6
(c)	66.7	100	80.0	64.2	68.3	66.2	71.3	100	83.3	50.8	75.0	60.6	80.0	80.0	80.0	64.2	68.3	66.2	91.9	61.3	73.6	76.7	51.1	61.4
(d)	66.7	100	80.0	74.6	71.4	73.0	71.3	100	83.3	58.6	75.0	65.8	80.0	80.0	80.0	74.6	71.4	73.0	91.9	61.3	73.6	78.4	52.3	62.8
(e)	91.7	91.7	91.7	76.1	92.7	83.6	79.0	79.0	79.0	65.0	72.5	68.5	70.0	70.0	70.0	58.2	70.9	63.9	86.5	86.5	86.5	85.8	85.8	85.8

Table 1: Comparison of the *LMetrics*(W_4) scores and the *OMetrics* scores.

for the same reason that LB^3 penalizes erroneous singleton clusters more than B^3 does.

In sum, the value of w_{sing} does not impact LB^3 and $LCEAF_e$. On the other hand, LMUC and $LCEAF_m$ pay more attention to singleton clusters as w_{sing} increases.

4.3 Impact of w_{nam}, w_{nom} and w_{pro}

When we were analyzing the *LMetrics* in the previous subsection, by setting w_{nam}, w_{nom}, and w_{pro} to the same value, we were not exploiting their capability to be linguistically aware. In this subsection, we investigate the impact of linguistic awareness using W_4 and W_5, which employ different values for the three weights.[9] To better understand the differences in recall and precision scores for each of the five system responses, we show these scores as computed by the *LMetrics* when they are used in combination with W_4.

First, consider response (a). As we can see from Figure 3 and the first row of Table 1, the *OMetrics* give decent scores to this output. Linguistically speaking, however, the system should be penalized more. The reason is that its output contributes little to understanding the document: in response (a), only the links between the PRONOUN mentions are established, and none of the PRONOUN or NOMINAL mentions is linked to a more informative mention that would enable the underlying entity to be inferred.

As expected, *LMetrics*(W_4) and *LMetrics*(W_5) assign much lower scores to response (a) than the *OMetrics*, owing to a relatively small value of w_{pro}. Also, we see that the *LMetrics*(W_5) scores are even lower than the *LMetrics*(W_4) scores. This suggests that the smaller the values of w_{pro} and w_{nom} are, the more heavily a resolver will be penalized for its failure to link a mention to a more informative coreferent mention.

Next, consider responses (b), (c) and (d). As the *OMetrics* ignore the type of mentions while scoring, they are unable to distinguish the differences among these three system responses: the *OMetrics* results in Figure 3 and their results in rows 2, 3 and 4 of Table 1 show that the scores for responses (b), (c) and (d) are identical. Linguistically speaking, however, they should not be. Response (d) contributes the most to document understanding, because the presence of NAME mention m_d^2 in its output enables one to infer the entity (*Jerusalem*) to which the mentions in S_{you} refer. In contrast, although response (b) correctly links S_{you} to PRONOUN mention m_f^2, one cannot infer the entity to which the mentions in S_{you} refer. The contribution of response (c) is in-between, because via m_e^2, we at least know that the mentions in S_{you} point to one *city*, although we do not know which *city* it is. Such differences in responses (b), (c) and (d) are captured by *LMetrics*(W_4) and *LMetrics*(W_5). Specifically, the *LMetrics* scores for response (d) are higher than those for response (c), which in turn are higher than those for response (b).

It is worth noting that the performance gaps between responses (b) and (c) and between responses (c) and (d) are larger under *LMetrics*(W_5) than under *LMetrics*(W_4). This is because w_{nom} and w_{pro} in W_5 are comparatively smaller. These results enable us to conclude that as the difference in the three edge weights becomes larger, the performance gap between a less informative resolver and a more informative resolver according to the *LMetrics* widens.

5 Conclusion

We addressed the problem of linguistic agnosticity in existing coreference evaluation metrics by proposing a framework that enables linguistic awareness to be incorporated into these metrics. While our experiments were performed on gold mentions, it is important to note that our linguistically aware metrics can be readily combined with, for example, Cai and Strube's (2010) method, so that they can be applied to system mentions.

[9]Like W_3, we set w_{sing} to 1 in W_4 and W_5, because this assignment makes $CEAF_m(W_3)$ rank response (e) above response (a), which we think is reasonable.

Acknowledgments

We thank the three anonymous reviewers for their detailed and insightful comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142. Any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of NSF.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the LREC Workshop on Linguistic Coreference*, page 563–566.

Jie Cai and Michael Strube. 2010. Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of the 11th Annual SIGDIAL meeting on Discourse and Dialogue*, pages 28–36.

Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32.

Sameer Pradhan, Lance Ramshaw, Mitchell Marcus, Martha Palmer, Ralph Weischedel, and Nianwen Xue. 2011. CoNLL-2011 shared task: Modeling unrestricted coreference in OntoNotes. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–27.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning: Shared Task*, pages 1–40.

Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand Index for coreference resolution. *Natural Language Engineering*, 17(4):485–510.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference*, pages 45–52.