› White Paper

# Discovering What You Want: Using Custom Entities in Text Mining

sas
THE POWER TO KNOW.

# Contents

A key element of data analysis is data discovery. Simply getting an idea of what is in your data – a "summary-level" view – can yield impressive insights.

But how does data discovery work with big data? Regardless of the size or type of data, discovery examines the data collection to provide new, meaningful insights. When it comes to deriving new, previously untapped business value from data, adding discovery to your analytics process adds a new level of knowledge beyond that seen in simple statistics.

After all, analytic discovery does more than review data. It critically assesses data. These well-established methods have deep roots in statistics, mathematics, computing science, linguistics and many other disciplines. Guided by domain experts and trained by the data, analytic discovery is a science that ensures that big data analysis yields results you can trust.

While text analytics is a somewhat newer form of analytics discovery, it is a powerful means to uncover information concealed in document collections. Text mining is the process of:

- Applying software technology to understand volumes of (unstructured) text.
- Analyzing the data to determine which terms are more prevalent than others.
- Learning how terms and phrases are related to one another.
- Understanding what the common themes in the document collection are.

One of the key challenges of text mining, however, is to use document representations that are both expressive (so they describe the collection in a meaningful way) while also being compact and succinct in describing the collection. Expressive representations retain at least some of the relational aspects of terms within a document. For example, you can understand that "fire station" and "station fire" mean different things, but if the representation discards the order of the terms, this distinction is lost.

The trouble with expressive representations is that while they help capture crucial contextual information, they can also introduce many unnecessary distinctions. For instance, in most text mining applications, there is no reason to distinguish "flaming fire" and "fire flaming." It is more useful to note that the terms individually occur.

Compact representations, such as a vector-based approach, only capture the information of each term independently, regardless of how they are associated with one another. With the right tools, and by thinking creatively, you can add features to these vector-based representations that can capture the desired relational and contextual information.

This paper demonstrates how to use a single but powerful feature in SAS® Text Miner to specify what relational features you want to capture. You still retain compact models, but they contain some contextual information that is targeted to your insight goal. Each relation becomes a new "term" in SAS Text Miner, explicitly driven by predefined pattern matching rules.

## Getting Started With Text Mining

Building good exploratory and predictive text models can be a challenge. You want to automatically discover themes and topics that are relevant and meaningful. And you hope that predictive text models are effective for classifying and predicting events. So when models fail to meet expectations, it's natural to try to improve performance with different controls and options available to tune the model. Sometimes, however, that's not successful, and the next step is to dig deeper to examine the representation of the documents themselves to try to improve underperforming models.

SAS Text Miner lets you easily analyze text data from the web, comment fields, books and other text sources. To do this, SAS Text Miner creates a "bag-of-words" vector representation of documents. A bag-of-words vector refers to a statistical approach based on term frequency, and not the context of the term, to drive the weighted importance of meaning. This model counts up how many times each word occurs in a document and stores that information in a vector where the position in the vector doesn't correspond to anything with the document.

SAS Text Miner does this by parsing sentences into terms, and these terms become variables used in further analysis.[1] Throughout the rest of this paper, the word "feature" describes terms that are used to represent documents. We use "feature" because variables in the document will not always be related to terms in the collection – but they may actually be a term that represents a relationship that exists in the document.

[1] Note that the parsed terms are implicitly applied, as the document-by-term matrix representation of terms is seldom created due to the sparsity of the generated table.

The properties of the parse node in SAS Text Miner allow you to control what features are used in your analysis. Options like part-of-speech tags, stop lists, entities, synonyms, spelling correction and stemming are also useful. But none of these settings can compare to the power and precision of linguistic feature creation – in defining specific features as well as the custom entities. This paper describes how you can develop and use custom entity features in discovery with SAS Text Miner to improve text analysis results.

## Document Representations for Text Mining

SAS Text Miner helps you discover new information, topics and term relationships that deepen your understanding of text information. How do you get started? First, you need to have a document collection to analyze.

After parsing the documents and identifying topics, the collection is numerically represented as a set of vectors that describe the text-mined patterns across the entire input document set. These patterns are an essential component to modeling. Without them, text mining cannot succeed. By choosing and creating distinctive linguistic features from the collection you are examining, you can improve the meaningful co-occurrence patterns between documents. And you can also find patterns based on those co-occurrences.

For example, consider the sentence shown in Table 1. This sentence has three different sets of features. Representation 1 shows only the term strings themselves. Representation 2 applies a stop list, stemming and tagging in the feature creation.

Representation 3 begins to provide some context to the text components. In this representation, relational information is encoded by creating a "baggage_fee_ent" term, which is a (defined) baggage fee entity. The system creates this entity because the sentence contains indicators that its author was using to describe how much money he or she spent to check bags.

The power of the approach used in Representation 3 centers on the creation of the term "baggage_fee_ent" – and how the system did this automatically. First you write a couple of linguistic rules in SAS Concept Creation for SAS Text Miner[2] to generalize $98 to a monetary concept. You can then relate this monetary concept to the term "bags." When you apply these rules in SAS Text Miner, the "baggage_fee_ent" term is created.

Because the string "$98" is unlikely to exactly match other monetary values (and even if it does, it might relate to some expense other than baggage fees), it's more important to generalize this as a monetary value and its relationship to bags. Exactly how, and to what extent, you customize your feature selection is tied to the goals of your analysis. In the previous example, do you want two documents to be more similar because they share exactly the same monetary value of "98?" Or do you want documents to be similar merely because they both mention any monetary value related to bags?

In the latter case, a generalization of the feature helps to map it to an established, or canonical, form. When one or more of the terms "bag," "bags" or "baggage" occur in your document collection, it's important to ask: Should these always increase the strength of relationship between any two documents that contain them? Alternatively, you may want to refine this further. For example, do you want the ability to distinguish between checking baggage and losing baggage? If so, you can create a feature that encodes a relationship when two or more terms are near each other in the same document.

| **Sentence** | **I was charged $98 for two bags.** | | | | | | |
|---|---|---|---|---|---|---|---|
| **Representation 1** | I | was | charged | $98 | for | two | bags |
| **Representation 2** | charge:Verb | $98:CURRENCY | two:Num | Bag:Noun | | | |
| **Representation 3** | charge:Verb | baggage_fee_ent | two:Num | | | | |

Table 1: Three representations of the same sentence.

[2] SAS Concept Creation for SAS Text Miner is an add-on technology that includes capabilities of SAS Enterprise Content Categorization, suited specifically for SAS Text Miner users who want to include custom entities in text mining analysis.

# What Are SAS® Text Miner Entities?

In SAS Text Miner, entities are terms (often multiword terms) that exist in a document and represent some predefined concept or class. These entities represent real-world elements such as a company, a person or a date. Entities become terms in a terms table and have a role that corresponds to the class that they belong to.

To extract an entity, the software must make a prediction about every term in the collection. For each term, a text mining classifier asks the question, "Is this the beginning of an entity of type X or not?" After the classifier finds the beginning, it makes another prediction about the end of the entity.

Sometimes, the classification is made in a deterministic way, which means it's based on the properties in the text (such as whether the first letter is capitalized or whether the term is in the predefined list of company names). Other times, the entity rule might be a match to a specific pattern, such as ***-***-****, which implies a North American phone number. These rules can be quite complex and can depend on part-of-speech tags and surrounding terms. In these cases, there may be a lot of uncertainty about which class, if any, a term might belong to.

Figure 1 shows the entity properties for text parsing. You can select the following entity settings:

- None: No entities will be detected.
- Standard: Uses the default set of entity rules, which are described in the next section.
- Custom: Enables you to specify the location of your own set of rules that you created in SAS Concept Creation for SAS Text Miner.
- All: Text parsing will extract both your custom entity types and standard entity types.

These entities are applied early in the parsing process. As a result, entities are discovered as features before applying synonyms and enforcing stop or start lists.

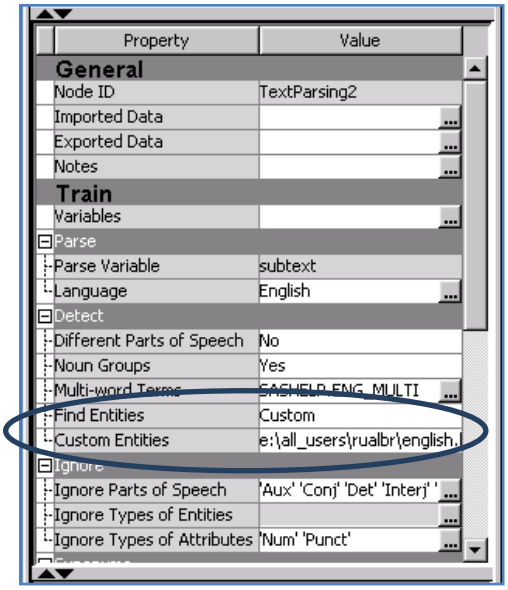


Figure 1. Entity properties in the text parsing node.

# Standard Entities in SAS® Text Miner

Standard entities can be detected by default in SAS Text Miner. All of the entity types listed below, except PROP_MISC, represent a common concept that might be applied in almost any domain. (The PROP_MISC entity type represents proper nouns in general that do not match any of the other types.) The complete list of the default standard entities for English language documents includes:

- Address
- Date
- Measure
- Person
- Social Security Number
- Title
- Company
- Internet
- Organization
- Phone
- Time
- Vehicle
- Currency
- Location
- Percent
- PROP_MISC
- TIME_PERIOD

SAS Text Miner finds these standard entities by using a preconfigured binary file previously created by SAS categorization technology. This binary file is accessed by the parsing procedure TGPARSE. Table 2 shows the output terms table from the example data, which contains some common entities.

| Obs | Term | Role | Attribute | Freq | numdocs | Keep | Key | Parent | Parent_ id | _ispar |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | $150.00 | CURRENCY | Entity | 1 | 1 | Y | 4 | | 4 | |
| 2 | 7 am | TIME | Entity | 1 | 1 | Y | 3 | | 3 | |
| 3 | john richards | PERSON | Entity | 1 | 1 | Y | 1 | | 1 | |
| 4 | san francisco | LOCATION | Entity | 1 | 1 | Y | 2 | | 2 | |

Table 2. Sample predefined entity output after parsing.

The primary purpose of the standard entities is to identify different classes of items. These classes provide a mechanism for you to further pursue entities by looking at documents that contain them. Then you can write your own SAS code to relate the entities. For example, you can explore how different company names and people's names relate to each other.

Standard entities are also useful in other modeling nodes. They can distinguish between two cases of the same term string used in different ways. In SAS Text Miner this becomes particularly important because it allows you to distinguish terms based on case, when the default is to put all terms into lowercase. You can also focus your analysis by choosing to include only specific entity types – or you can manually treat all entity types to be a synonym of the same term.

## Custom Entities in SAS® Text Miner

The custom entity feature of SAS Text Miner inputs a file created in SAS Concept Creation for SAS Text Miner that allows you to control the features that you want to identify during text parsing. You can define custom entities to discover items that belong to some new class for your domain.

In the case of the airline data, you may be interested in creating an airline entity or an airport entity. After you decide which linguistic properties define these things, you can identify them, use them in reports and distinguish between different types of entities.

But you rarely want to stop there. Custom entities don't need to be interpreted in the same way as standard entities; they aren't limited to a representation of only real-world elements in your text. Instead, custom entities allow you to become creative by using specific, helpful information extracted from the text that you can use as features. Ultimately, custom entities help you:

- Identify elements that represent relationships between multiple terms.
- Detect co-references.
- Create general pattern matches of specific linguistic elements in the text.
- Build increasingly complex rules for extraction that are based on earlier rules.

All of this will help you create useful features for model building.

Let's take a look at a pair of complementary perspectives on custom entities. First, let's consider a type of programmatic synonym list. A custom entity is written as a generalized variety of distinct terms that have some type of organized form. In that form, this custom entity will match across documents. When the custom entity is encoded into a feature, it can also indicate which documents are similar to each other.

A second type of custom entity allows you to capitalize on relational elements between terms – when that relation can be used to create a new feature (that may simply be a refinement). Together, the two perspectives (entities as synonyms and entities as relational features) are more powerful than either would be in isolation.

## Custom Entities as Synonyms

Entities can provide a complex but powerful approach to synonym discovery and assignment when all entities of a given type are mapped to a canonical term. When you use custom entities, you control what types of text strings you would like to treat as synonyms. Unlike the standard entities in SAS Text Miner, those developed with SAS Concept Creation for SAS Text Miner allow you to generalize entities to match broader patterns – so you identify more synonyms as a result.

For example, consider airport codes, such as SFO for San Francisco International Airport. Suppose you do not have a list of all airport entities, but you want to use the pattern of three uppercase letters. The following syntax in SAS Concept Creation for SAS Text Miner allows you to extract all terms of this type:

Top: AIRPORT_CODE:  REGEX:[A-Z][A-Z][A-Z]

This syntax might be too general. It would also identify random instances of uppercase terms (such as "AND") in the text documents. Now suppose you also want to view documents as similar as long as they mention any airport code. Since SAS Text Miner uses the term-role pair[3] to distinguish terms, finding two airport codes in two different documents, such as D1 and D2, shown in Table 3, doesn't create any relationships between the documents.

However, you can create a relationship if you represent the term with a parent term that is formed from the role. In Table 4 the two terms are represented by the same parent term, AIRPORT_CODE_ENT, because they share the same entity type. In the Text Filter node, you can manually assign all terms that have the AIRPORT_CODE role to a single representative term.

To accomplish this, you specify a macro variable at the beginning of the code for the roles you want to be mapped to a canonical form as a parent. The macro adjusts the terms table and the term-document frequency table to account for the parents that are introduced. A third alternative is to use the Text Topic node to create a user-defined topic.

As mentioned previously, entities typically represent a real-world concept, such as a company name or a person's name. By using the custom entity property, you can assign a flexible meaning to entities that allows you to capture additional features.

In the airline industry, there are terms that have special meanings in a given context. Being "bumped" often means that the airline overbooked the flight, moving passengers to the next flight. This is clearly a different meaning from being physically "bumped" by a passenger while you are stowing your bag.

You can use context and relationship between terms to capture this meaning and distinguish between the two uses. The following custom entity (represented as a linguistic rule) can help you capture the cases in which the text means being bumped from a flight:

BUMPED_FLIGHT: CONCEPT_RULE:(ORDDIST_10, "_c{bumped}", (OR, "flight", "airplane", "plane")).

For longer documents, it's better to create an entity that encodes terms appearing near one another. Imagine a long complaint written to an airline that covers everything from the service to the food to the price of airfare. Early in the document, the complaint is about "narrow-minded employees," while later the complaint is that the "seat was dirty." The bag-of-words approach to representing your document correlates the term "narrow" with the term "seat" as much as it does with "minded."

| Document | Term | Role | Parent | Parent Role |
|---|---|---|---|---|
| D1 | SFO | AIRPORT_CODE | | |
| D2 | LGA | AIRPORT_CODE | | |

Table 3. Airport codes without synonyms.

| Document | Term | Role | Parent | Parent Role |
|---|---|---|---|---|
| D1 | SFO | AIRPORT_CODE | AIRPORT_CODE_ENT | AIRPORT_CODE |
| D2 | LGA | AIRPORT_CODE | AIRPORT_CODE_ENT | AIRPORT_CODE |

Table 4. Airport codes treated as synonyms.

[3] A term-role pair refers to the relationship between a specific term and its use in the document.

Although the previous examples are for specific cases, there are linguistic aspects that can be discerned in the analysis. You can use many linguistic rules for sentiment, which can be also be automatically included in the model. This could include rules to capture the meaning of a term when it is reversed by the use of "not" or "didn't."

When using SAS Concept Creation for SAS Text Miner rules, you don't need to stop associating actual terms in your document to create a new term. You can refer to earlier rules to make increasingly powerful, complex and nested rules. In the earlier example, after you define an entity for lost baggage, you can relate that to an entity for flight delay.

For more details on how to create custom entities for SAS Text Miner, view this technical paper: support.sas.com/resources/papers/proceedings13/100-2013.pdf

## Summary

Custom entities give you context-sensitive control over your text mining projects by allowing you to introduce your own subject matter experience to an otherwise machine-learned, statistical discovery approach. The custom features that you introduce can substantially alter the model performance by creating specific co-occurrence patterns that the bag-of-words model could not otherwise find.

In most cases, custom entities initially expand the number of distinct features in your model. To improve results even further, you can try treating all entities of the same type as if they were the same feature. Additionally, you are likely to gain more from custom entities that are applied to longer documents than to short documents. Custom entities are effective when relating terms that are near one another. The longer your documents, the more this applies.

Other techniques can be used to analyze text files in different ways. For example, a model can show two documents are similar, not only because of the co-occurrence patterns between those two documents, but also because of the co-occurrence patterns among those two documents and to other documents. This means that you can create custom entities that have a significant effect throughout the collection.

The custom entity feature in SAS Text Miner, along with the SAS Concept Creation for SAS Text Miner package, gives you the power and control to manipulate model building to add more real-world, contextual meaning into the text analytics process. As a result, you can more accurately derive even more value from text documents – and begin to move from discovery to realizing precisely what you need from your unstructured data.

Custom entities give you context-sensitive control over your text mining projects by allowing you to introduce your own subject matter experience to an otherwise machine-learned, statistical discovery approach. The custom features that you introduce can substantially alter the model performance by creating specific co-occurrence patterns that the bag-of-words model could not otherwise find.

To contact your local SAS office, please visit: sas.com/offices



107347_S124628.1014

THE POWER TO KNOW.