# Introducing prose v2.0.0: Bringing NLP to Go 介绍散文2.0.0版：实现NLP

A guide to using Go for natural language processing (NLP).  
使用Go进行自然语言处理（NLP）的指南。

We’re pleased to announce the of prose, a natural language processing (NLP) library for Go.  
我们很高兴地宣布散文，一个自然语言处理（NLP）图书馆为Go。

v2.0.0 represents a major shift in the project’s focus: instead of simply offering an assortment of prose-related utilities, we’re focusing on bringing a more refined NLP experience to Go. This means that the development of v1.0.0’s higher-level features (e.g., the title-case converter) will be moved to other repositories going forward.  
v2.0.0代表了项目重点的一个重大转变：我们不是简单地提供各种与散文相关的实用程序，而是专注于带来更精致的NLP体验。这意味着v1.0.0的高级特性（例如，标题-案例转换器）的开发将被转移到其他存储库中。

In order to avoid breaking code already importing prose, v2.0.0 will be exposed via gopkg.in/jdkato/prose.v2— allowing github.com/jdkato/prose to still point to v1.0.0.  
为了避免破坏已经导入散文的代码，v2.0.0将通过gopkg.In/jdkato/prose.v2公开-允许github.com/jdkato/prose仍然指向v1.0.0。

Among the new features of v2.0.0 is a new, more cohesive API built around Documents.  
在v2.0.0的新特性中，有一个围绕文档构建的更具内聚性的新API。

The document-creation process consists of four steps — tokenization, segmentation, POS tagging, and named-entity extraction — which are discussed in more detail below.  
文档创建过程包括四个步骤 - 标记化、分段、POS标记和命名实体提取 - ，下面将详细讨论这些步骤。

### Tokenization 符号化

Given a piece of text, tokenization is the task of breaking it up into units referred to as tokens. For example,  
给定一段文本，标记化是将其分解为称为标记的单元的任务。例如，

And while there’s really no “correct” way to tokenize text, you definitely need to do more than identify word boundaries to be useful. Some examples of non-word tokens that prose can identify are given below.  
虽然没有什么“正确”的方法来标记文本，但你绝对需要做的不仅仅是识别单词边界才有用。下面给出一些散文可以识别的非单词标记的示例。

So, for example, a sentence like  
例如，一个句子

becomes  
变成

### Segmentation 分割

Text segmentation is the process of dividing text into sentences. This is generally a more challenging task than tokenization due to the ambiguity of sentence boundaries. Fortunately, the developers of the have complied a that can be used to evaluate segmenters. Their results are as follows (with prose added):  
文本分割是将文本分割成句子的过程。由于句子边界的模糊性，这通常比标记化更具挑战性。幸运的是，开发人员已经编写了可用于评估分段器的。结果如下（加上散文）：

As you can see, prose performed relatively well. Most of its missed cases (Golden Rules 31-39) were list-containing sentences, which seem to be pretty rare.  
如你所见，散文表现得相对较好。它漏掉的大多数案例（黄金规则31-39）都是包含句子的列表，这似乎非常罕见。

### Part-of-Speech (POS) Tagging 词性标注

POS tagging is the process of assigning part-of-speech tags (e.g., NN for nouns) to individual tokens. prose includes a POS tagger based on Matthew Honnibal’s .  
词性标记是将词性标记的一部分（如名词的NN）分配给单个标记的过程。散文包括一个基于马修本尼巴尔的POS标记。

To evaluate the performance of our tagger, we used the portion of the University of Pennsylvania’s available through Python’s library ( for the test script).  
为了评估标记器的性能，我们使用了Python库中宾夕法尼亚大学的可用部分（用于测试脚本）。

† Given a list of reference values and a corresponding list of test values, return the fraction of corresponding values that are equal.  
给参考值列表和相应的测试值列表，返回相等的相应值的分数。

source: NLTK  
来源：NLTK

### Named-Entity Recognition (NER) 命名实体识别（NER）

NER is the process of assigning labels to particular entities within text (e.g., people, places, organizations, etc.). v2.0.0 includes a much improved version of v1.0.0’s chunk package, which can identify people (PERSON) and geographical/political Entities (GPE) by default.  
NER是将标签分配给文本中特定实体（例如，人、地点、组织等）的过程。v2.0.0包含了v1.0.0的chunk包的一个改进版本，默认情况下，它可以识别人（PERSON）和地理/政治实体（GPE）。

This generally works pretty well. However, instead of focusing on fine-tuning the default model, we’ve put a lot of effort into making it easy to train your own models for specific use cases — for instance, maybe you want to be able to identify all Apple products as APPLE.  
通常情况下，这种方法效果很好。然而，我们并没有专注于对默认模型进行微调，而是花了大量精力使您能够轻松地为特定的用例（例如，您可能希望能够将所有苹果产品都识别为苹果）培训自己的模型。

To train a new model, all you need to do is provide a slice of LabeledEntities:  
要培训新模型，您只需提供一个标签：

Keep a look out for our next post, which will cover training a new prose-compatible NER model using .  
留心我们的下一篇文章，这篇文章将介绍如何使用新的散文兼容的NER模型。

### Going Forward 前进

prose started out as simply an assortment of prose-related utilities we needed for , however, its goals are now more inline with what JavaScript’s has accomplished: being a relatively simple, yet practical NLP library.  
散文最初只是我们所需要的与散文相关的各种实用工具，然而，现在它的目标更符合JavaScript的成就：成为一个相对简单但实用的NLP库。

The next major step for the project is to add support for text classification, which will allow us to label text as being related to certain topics. If you’d like to get involved, head over to the [GitHub repository](https://github.com/jdkato/prose).