Simple Bootstrapping Algorithm (pseudocode)

The SimpleBoot bootstrapping algorithm is a simple Know-It-All-style relation learner for unary relations such as IsNation(x) and IsConference(x). Initially, the algorithm is primed with a set of seed rules (i.e., phrases) that can be used to extract instances of the target relation. For instance, in the case of IsNation(x), a seed rule might be "`nations such as`" and by searching the web for this phrase, the algorithm might find instances such as "`China`" and "`India`."
Then, the algorithm tries to find new rules with which to search for even more instances) by learning them from the current instances. It does so by taking a subset of the current instances, searching the web for them, and looking for common phrases that appear either before or after multiple instances. The assumption is made that these phrases often appear in conjunction with instances of the target relation, and they are used to extract more instances. This process repeats until all rules are exhausted.

```
procedure SimpleBoot(relation, seedrules)
    | Step 0: Create and initialize active learner / probability store. Add the seed rules. Ac-
    | commodate the multi-relation case with an argument that specifies which relation the rule
    | should extract.
    A = NewActiveLearner(relation, unary)
    for all rule ∈ seedrules do
        A.AddRule(rule, relation, seed)
    end for
    while |unsearched rules| > 0 do
        | Step 1: Get a ranked list of unsearched rules that the active learner thinks ought to extract
        | instances of the target relation with high probability. Use the top five rules in this round
        | of extractions.
        rules = (A.GetRankedRules(relation) ∩ unsearched rules) [1, . . . , 5]
        for all rule ∈ rules do
            new instances = ExtractNewInstances(r)
            for all inst ∈ new instances do
                count = # times rule extracted inst
                A.AddInstance(inst, count, rule)
            end for
            unsearched instances = unsearched instances ∪ new instances
        end for
        unsearched rules = unsearched rules ∖ rules
        | Step 2: Get a list of unsearched instances that the active learner thinks are instances of
        | the target relation with high probability. Search for all the instances and extract the five
        | words to the left and right of each. Find sequences of two or more words that appear
        | across multiple instances, and add them as potential new extraction rules.
        instances = (A.GetRankedInstances(relation) ∩ unsearched instances) [1, . . . , 5]
        new rules = ExtractNewRules(instances)
        for all rule ∈ new rules do
            A.AddRule(rule, relation, instances)
        end for
        unsearched rules = unsearched rules ∪ new rules
        unsearched instances = unsearched instances ∖ instances
    end while
    | Step 3: Get a list of all instances that have a high probability of being instances of the
    | target relation and record them along with their probabilities.
    instances = A.GetRankedInstances(relation)
    for all inst ∈ instances do
        p = A.GetInstanceProbability(inst, relation)
        if p > 1/2 then
            RecordInstance(inst, p, relation)
        end if
    end for
end procedure
```
