Parameter estimation under uncertainty with Simulated Annealing applied to an ant colony based probabilistic WSD algorithm

Andon TCHECHMEDJIEV[1] *Didier SCHWAB*[1] *Jérôme GOULIAN*[1]
Gilles SÉRASSET[1]
(1) Univ. Grenoble-Alpes, LIG-GETALP, France
{andon.tchechmedjiev, didier.schwab, jerome.goulian, gilles.serasset}@imag.fr

ABSTRACT
In this article we propose a method based on simulated annealing for the parameter estimation of probabilistic algorithms, where the solution provided by the algorithm can vary from execution to execution. Such algorithms are often very interesting to solve complex combinatorial problems, yet they involve many parameters that can be difficult to estimate manually due to their randomized output. We applied and evaluated a method for the parameter estimation of such algorithms and applied it for an Ant Colony Algorithm for WSD. For the evaluation, we used the Semeval 2007 Task 7 corpus. We split the corpus and took in turn one text as a training corpus and the four remaining texts as a test corpus. We tuned the parameters with an increasing number of sentences from the training text in order to estimate the quantity of data necessary to obtain an efficient and general set of parameters. We found that the results greatly depend on the nature of the text, even a very small amount of training sentences can lead to good results if the text has the right properties.

RÉSUMÉ (French)
Estimation de paramètres à base de Recuit Simulé sous incertitude appliquée à un algorithmes à colonies de fourmis probabiliste.
Nous proposons une méthode basée sur un Recuit Simulé pour l'estimation de paramètres pour des algorithmes probabilistes où les solutions générées varient. Ces algorithmes sont souvent très intéressant pour la résolution de problèmes combinatoires complexes, mais ils requièrent de nombreux paramètres pouvant être difficiles à estimer manuellement à cause de la nature aléatoire des solutions. Nous avons appliquésPlus spécifiquement, nous appliquons et évaluons cette méthode à pour estimer les paramètres de tels algorimes et l'appliquons à un Algorithme à Colonies de Fourmis pour la désambiguïsation lexicale. Pour l'évaluation, nous avons utilisé le corpus de Semeval 2007 Tâche 7. Nous avons repectivement séparé un texte comme corpus d'entraînement et les quatre autres comme corpus de test. Nous estimons les paramètres pour un nombre croissant de phrases pour déterminer combien de données sont nécessaires pour obtenir un ensemble de valeurs de paramètres générales et efficaces. Nous concluons que la qualité des résultats dépends de la nature des textes. Même des petites quantités de de phrases peuvent suffir à obtenir de bons résultats, du moment que le texte à les bonnes propriètés.

KEYWORDS: Word Sense Disambiguation, Parameter Estimation, Simulated Annealing, Uncertainty, Stochastic Algorithms.

KEYWORDS IN FR: Désabiguïsation Lexicale, Estimation de Paramètres, Recuit Simulé, Incertitude, Algorithmes Stocastiques.

Proceedings of the First International Workshop on Optimization Techniques for Human Language Technology, pages 109–124,
COLING 2012, Mumbai, December 2012.

1 Introduction

Word Sense Disambiguation (WSD) is a very difficult yet central task in Natural Language Processing (NLP), that pertains to the labelling of the words of a text with the senses that most closely match the context. Numerous approaches exist to tackle WSD, yet many of these methods have in common a sizeable complexity, reflected by a number of parameters that need to be tuned in order to obtain the best performance. Depending on the number of parameters, it can be very difficult for a human to directly estimate the optimal parameters. Even then, it remains a question of guesswork with no guarantee of success.

This issue is all the more salient with algorithms and models that typically involve many parameters upon which the results greatly depend. Such algorithms for WSD include Neural Networks (Veronis and Ide, 1990), Simulated Annealing (SA) (Cowie et al., 1992), Genetic Algorithms (GA) (Gelbukh et al., 2003) and Ant Colony Algorithms (ACA) (Schwab and Guillaume, 2011), and many other machine learning methods.

In such approaches, the values of the parameters are paramount. They are what make the algorithms work for specific applications such as WSD. Yet, the authors of the aforementioned articles give little insight as to how the parameters are determined. They only provide the "best" values. Although a *trial and error* approach is a good start when devising an algorithm or tailoring it to a new application, it makes the adaptation and the scalability of such systems an issue. Naturally, for well known and understood algorithms such as Simulated Annealing, there are some theoretical criteria to select the parameters. However, the process remains to be repeated for every new setting and can be tedious to perform.

Furthermore, even with few parameters, evaluating all possible parameter configurations is a prohibitively long process. For example, in the case of a stochastic algorithm with 10 parameters that have 20 possible values each, assuming the algorithm needs to be run 100 times (due to its stochastic nature) and that one execution takes on average 30 seconds, the time necessary to evaluate all parameter combinations is $20^{12} \cdot 100 \times 30 = 3 \times 10^{15} s$ which is almost 1 billion years! Thus, the advantages of considering automated or adaptive heuristic approaches to the estimation of parameters are clear.

In the case of a deterministic algorithm, many combinatorial optimisation methods can be used with little effort. However, due to the complexity of WSD, many approaches are statistical or probabilistic in nature and the use classical combinatorial optimization heuristics becomes impossible. Therefore, we adapt the classical simulated annealing algorithm to work for an uncertain objective function based on the ideas of (Painton and Diwekar, 1995), by using standard non-parametric statistical significance tests.

First, we present the tools and the metrics for the evaluation of WSD, followed by the description of the Ant Colony Algorithm considered in this article. Then, we briefly survey other approaches for parameter estimation under uncertainty and express the problem formally. Subsequently, we present the general formulation or simulated annealing and different aspects pertaining to it and then present its adaptation to uncertain objective function. We then present and analyse our experimental protocol and the results of the experiments. Finally, we conclude on the results and draw some perspectives for future research.

2 Evaluation of WSD Systems

The use of Precision and Recall constitutes the standard evaluation metrics for WSD systems.

Precision represents the number of correctly disambiguated words among all the attempted words, while Recall represents the number of correctly disambiguated words among all the ambiguous words in the document. It is customary to compute the F1 score between Precision and Recall as $\frac{2 \cdot P \cdot R}{P+R}$, in order to have a single evaluation metric that equally captures the information conveyed by both Precision and Recall.

In this work in particular we use the Semeval 2007 Task 7 all words coarse grained corpus for the evaluation of the quality of the disambiguation. The all-words task provides a corpus of texts, where in each text all ambiguous words (T) need to be disambiguated by tagging them with Wordnet 2.1 sense

keys. An algorithm must correctly tag as many ambiguous words as possible (TP) depending on the context. A sense is considered correct if the sense key assigned to it matches the sense provided in the gold standard or any of its subsumed senses (hence coarse-grained). If we write the number of incorrectly assigned senses as (TN), then, for this particular task, precision can be expressed as $P = \frac{TP}{TP+TN}$ and recall as $R = \frac{TP}{T}$

The choice of the all words task in particular is based on the functioning of the algorithm that is more adapted to a globally coherent text, rather than independent sentences or words. As such, the metrics available for the evaluation are Precision, Recall and F-measure, which leads to some important restrictions on the methods or criteria available for parameter estimation as will be detailed in the subsequent sections.

3 Context: an ant colony algorithm for WSD

The ant colony algorithm presented by (Schwab and Guillaume, 2011) and (Schwab et al., 2012) that our work focuses on, offers some interesting properties regarding the quality of the solutions, which are on par with state-of-the-art knowledge rich WSD systems, combined with a fast execution time compared to the latter systems. Furthermore, it is also well suited to working on full texts at a time with a full annotation coverage.

Moreover, its stochastic nature leads to the generation of solutions with variable precision scores, which is usually not a very sought after property in algorithms. However, this variability enables the use of voting strategies that lead to results approaching supervised WSD systems and overcome the elusive (for unsupervised and knowledge-based systems) First Sense Baseline. Due to the quick execution time (in the 60s to 65s range), voting strategies can directly be applied while still leading to cumulative execution times for several executions that are lower to that of other systems.

However, those qualities are also a major disadvantage when it comes to selecting and tuning the parameters of the algorithm. Not only are there relatively many, but they can only be determined manually to some extent through painstaking trial and error modifications, with no guarantees of optimality. Notwithstanding with the fact that the stochastic nature of the algorithm makes it rather impractical to determine whether the improvement resulting from a given parameter change is simply due to random variations of the solution distribution or really to significant variations.

3.1 Parameter model

Let us now review the various parameters of the system, their typical value ranges and meaningful increments. The parameters are summarized in Table 2. There are seven parameters in total, five of which are discrete and represented by integers and two continuous parameters represented as positive real numbers between zero and one.

Given the number of parameters, even with some knowledge about how the algorithm works, one can at best chose sensible parameter value ranges in order to limit the combinatorial explosion.

Initially, before the estimation method proposed in this paper, we determined the values of some parameters by making iterative and independent changes. Of course such an approach is limited due to the fact that it is a heuristic based on the assumption that the parameters of the system are independent. For ACA it is in fact the opposite, as it is with other methods.

The values obtained through this process were $\omega = 10$, $E_a = 1$, $E_{max} = 8$, $E_0 = 10$, $\delta_v = 0.9$, $\delta = 0.9$, $L_V = 90$ and yielded results around 75% on the Semeval 2007 Task 7 corpus (Schwab and Guillaume, 2011).

Furthermore, given the number of parameters and their value ranges, an exhaustive enumeration is intractable. If we calculate the number of combinations (assuming 0.01 steps for continuous variables), we obtain $60 \times 60 \times 100 \times 55 \times 25 \times 35 \times 100 = 17325 \cdot 10^8$ combinations. Knowing that due to the stochastic nature we may need to make at least 50 or 100 executions, we get to $17325 \cdot 10^9$ combinations.

Notation	Description	Value	Exploration granularity
E_a	Energy taken by an ant when it arrives on a node	1–60	1
E_{max}	Maximum quantity of energy an ant can carry	1–60	1
δ_ϕ	Evaporation rate of the pheromone between two cycles	0.0–1.0	0.01
E_0	Initial quantity of energy on each node	5–60	1
ω	Ant life-span	5–30 (cycles)	1
L_V	Odour vector length	20–200	5
δ_V	Ratio of odour vector components (words) deposited by an ant when it arrives on a node	0.0–1.0	0.01

Table 2: Parameters of the Ant Colony Algorithm and their typical value-ranges

Assuming leaps in computational power or heavy parallelism that would make the algorithm take only one second to run, the search for the optimal parameters would still take 549372 years.

For this reason, we are interested in finding and automated way of estimating the *optimal* parameters. Of course, when dealing with linguistic data, there is no such thing as *optimal*. By *optimal*, we merely mean a set of parameters that yield results with F-scores as high as can be achieved.

4 Related work

In the context of this article, when considering WSD algorithm evaluated as described in Section 2, the output of the evaluation of the algorithm output is simply a F-score percentage. In order to apply classical estimation theory approaches (Kay, 1993), we would need to either find a model of the posterior PDF, or to empirically estimate it. However, given the size of the search space and the cost of running the algorithm, this would be very much equivalent to making an exhaustive search. Furthermore, the relationship between the values of the parameters and the resulting scores are non-linear. However, there are some models that attempt to provide a probability distribution for precision, recall, and F-measure for information retrieval that could potentially be applied to WSD (Goutte and Gaussier, 2005).

In fact, it is much simpler to treat the estimation as a combinatorial optimisation problem, by attempting to simply maximize the f-score value. In this context, given that we want the ability to handle many parameters that can take many discrete values, we need to consider heuristics for an efficient estimation of complex non-linear multivariate functions.

A popular choices for parameter estimation are Genetic Algorithms (GA) or Simulated annealing (SA) among others. For instance, GAs have been widely applied, either in a general parameter estimation context, such as in (Sharman and McClurkin, 1989), (Yao and Sethares, 1994) or (Paterakis et al., 1998); or for specific application domains such as robotics (Bolhasani, 2004), applied statistics (Pan et al., 1995), meteorology (Lee et al., 2006), chemistry (Katare et al., 2004) and many others.

For WSD, (Daelemans et al., 2003) and (Decadt et al., 2004) have also proposed to use GAs for joint parameter estimation and feature selection. Although joint optimisation is very interesting, it is application and task specific, and would be difficult to implement in a probabilistic setting.

All the techniques mentioned above are meant to optimise a deterministic objective function. When the objective function itself is uncertain or noisy, instead of a single value, one has a set of observation samples with different values that have to be dealt with. The main issues are the question of how to know what value to consider for the maximisation and how to take the variability into account to avoid effects due to random chance.

One model of *stochastic* simulated annealing has been proposed in (Painton and Diwekar, 1995) and then further extended in (Kim and Diwekar, 2002), where the evaluation function is sampled several times, and

the expected value is incorporated into a modified objective function, with the variance acting a penalty term.

5 Problem formalization

Before describing the parameter estimation algorithm, it is important to formulate in a more formal and generic way the parameter model of any WSD algorithm with a randomized output.

Let $\vec{\theta} = [\theta_1, \theta_2, \ldots, \theta_i, \ldots \theta_n] \in \Theta$, be the parameter model of a WSD algorithm, where $\Theta = \Theta_1 \times \Theta_2 \times \ldots \times \Theta_i \times \ldots \times \Theta_n$ and each Θ_i is a finite discrete set of integers or a bounded real range.

For example, in the case of the parameter model of the ant colony algorithm, the general form of θ would be $\theta = \{\omega,\ E_a,\ E_{max},\ E_0,\ \delta_v, \delta,\ L_V\}$. The initial set of parameters would be an instance written as $\theta_I = \{10, 1, 8, 10, 0.9, 0.9, 90\}$.

We can represent a deterministic WSD system with a function WSD_C, that for a given corpus C and a parameter vector $\vec{\theta}$[1] returns a $F-score$ value between 0 and 1:

$$WSD_C : \Theta \rightarrow 0 \leq x \in \mathbb{R} \leq 1 \tag{1}$$

$$\theta \mapsto F_{score} \tag{2}$$

Thus, the problem of finding the best θ, θ^* can be formalized as follows:

$$\theta^* = \arg\max_{\theta \in \Theta} WSD_C(\theta) \tag{3}$$

In other words, if we define a sequence θ_n that enumerates possible parameter combinations in Θ, the problem can be expressed as:

$$\theta_n^* = \begin{cases} \theta_n & \text{if } WSD_C(\theta_n) > WSD_C(\theta_{n-1}) \\ \theta_{n-1} & \text{otherwise} \end{cases} \tag{4}$$

However, when the WSD system is probabilistic, there isn't a unique F-score given for a θ_n, but every evaluation of WSD_C may potentially return a different F-score value. Thus, WSD_C follows an unknown probability distribution X that depends on θ_n: $WSD_C(\theta_n) \sim X(\theta_n)$.

A sensible approach in this case would be to follow the model (Fig. 1b) proposed by (Painton and Diwekar, 1995), and to evaluate the objective function WSD_C several times and then consider the expected value as an estimator of the resulting distribution. However, even if the means are different, there is no guarantee the difference is not due to random chance.

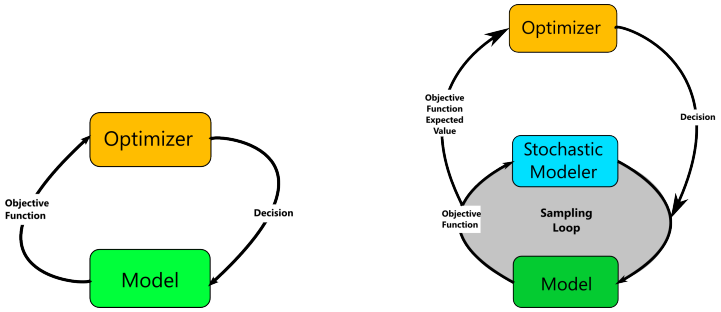
The more general way to deal with this uncertainty about the statistical significance of the difference of expected values is to simply apply a standard statistical test in order to determine the significance level and decide whether or not to go ahead with the comparison. Another simpler approach, the one retained in fact, is to integrate into the scoring function a penalty factor that directly takes into account the variability of the distributions. This problem will specifically be addressed in more detail in Section 6.3.

6 Simulated Annealing

6.1 General Presentation

Simulated annealing is a stochastic combinatorial optimisation technique originally proposed by (Kirkpatrick et al., 1983). Given a function $f(\theta)$ that takes a vector of discrete parameters θ, Simulated

[1] From now on, the vector arrow will be omitted and $\vec{\theta}$ will be written as θ.



(a) Classical optimisation model (b) Stochastic optimisation model

Figure 1: Classical optimisation model versus stochastic optimisation model

Annealing aims at finding the combination of parameter that maximises or minimises that function. (Kirkpatrick et al., 1983) draw a parallel between statistical mechanics and combinatorial optimisation and apply ideas from the Metropolis-Hastings algorithm (Metropolis et al., 1953) (simulation of the behaviour of many body systems) to combinatorial optimisation.

The principle of Simulated annealing is to first start from an initial configuration (combination) of the function parameters θ_0 as well as an initial temperature T_0. The initial θ_0, is often chosen randomly, but putting an already good solution can accelerate the search.

Then, the algorithm iteratively applies a random change operator R to the current configuration in order to obtain a modified configuration $\theta' = R(\theta)$.

The value of the modified configuration $f(\theta')$ is compared to the value of the current configuration $f(\theta)$, such that $\Delta f = f(\theta') - f(\theta)$. An acceptance function $P(\Delta f)$ is then used to determine whether to keep the current configuration or to accept the modified configuration.

In the original SA algorithm, the acceptance function is expressed using the Metropolis criterion and the Boltzmann distribution as shown in Equation 5.

$$P(\Delta f) = \begin{cases} 1 & \text{if } \Delta f < 0 \\ exp\left(\frac{-\Delta f}{T}\right) & \text{otherwise} \end{cases} \tag{5}$$

With gradient descent or hill climbing search, only configuration that have a higher score are kept, while inferior configurations are systematically rejected. The problems with these approaches lies in the fact that complex functions often have many local minima and maxima, and the algorithm will likely converge on such a minimum or maximum.

Simulated Annealing, on the other hand, has a probability of accepting inferior configurations as a local minima/maxima escape strategy. The initial temperature is chosen so that at the beginning of the exploration, inferior configurations are almost always kept. After each iteration of the simulation, the temperature decreases and with it the probability to keep inferior configuration. As such, the algorithm starts with a wide and coarse exploration of the search space and then gradually converges on a fine grained exploration of a specific area around a minimum or maximum (hopefully close to the global minimum or maximum).

The convergence criterion for the algorithm can either be when the temperature reaches a threshold called

freezing temperature or when the system is deemed to stabilize (usually a number of cycles during which improvements to the objective function are marginal or non-existent).

6.2 Cooling schedule and candidate generation

In SA, the way the temperature decreases is key to the performance of the algorithm. A logarithmic cooling schedule ($T_n = \frac{T_0}{ln(n)}$) is theoretically sufficient in order to guarantee convergence (Ingber, 1996), however, it is too slow for many problems. A more common cooling schedule is geometric $T_n = T_0 \cdot (\alpha)^n$, where alpha is the cooling factor. However, on the one hand the slope of the cooling curve will be very steep even for α close to one and on the other hand, the curve itself is convex. This means that the slope of the curve will get steep very fast, thus leading to a fast convergence at the cost of optimality.

For this reason, (Ingber, 1996) proposes in his Adaptive Simulated Annealing algorithm to use an inverse exponential cooling schedule that will decrease quickly at the start and then slow down. Such a schedule corresponds much better to combinatorial optimisation algorithms, where we want to start with a broad search (accepting many inferior configurations) and then focus on a specific area of the search space. (Ingber, 1996) proposed the expression in equations 6 and 7, which we adopted for our implementation. Here, m and n are two scaling factors that can be used to tune the schedule for specific problems.

$$T_k = T_0 \cdot exp\left(-c_i \cdot k^{\frac{1}{|\theta|}}\right) \tag{6}$$

$$c_i = m \cdot exp\left(\frac{-n}{D}\right) \tag{7}$$

Furthermore, the traditional SA, generates a new candidate configuration by uniformly selecting an index of the vector and by applying a random uniform change on the value, over the full range of possible values for that index. (Ingber, 1996) argues that once the system is colder, there is no point to explore the full range. Indeed, exploring an increasingly smaller neighbourhood as the temperature lowers and the systems converges on a specific area of the search space, allows an increase in computational efficiency while having no negative effects on the end result.

Equation 8 presents the probability distribution used for the candidate generation for each parameter, where $u_i \in U(0,1)$ is a uniform random number between 0 and 1.

$$y = sign\left(u - \frac{1}{2}\right) T_k \left[\left(1 + \frac{1}{T_k}\right)^{|2u-1|} - 1\right] \tag{8}$$

From there, the new value of a parameter $\theta^{(i)}$ of θ at iteration k can be determined with the expression in Equation 9 for a real parameter and Equation 10 for an integer parameter.

$$\theta^{(i)}_{k+1} = \theta^{(i)}_k + y(\theta^{(i)}_{max} - \theta^{(i)}_{min}) \tag{9}$$

$$\theta^{(i)}_{k+1} = \theta^{(i)}_k + \lfloor y(\theta^{(i)}_{max} - \theta^{(i)}_{min}) \rfloor \tag{10}$$

6.3 Handling uncertainty

As mentioned in Section 5, when dealing with an uncertain objective function, one needs to add an extra loop in the process, in order to generate samples from the evaluation of the objective function f and then to use the expected value as a global objective function.

For a given θ, we can build a list of samples $Ls(\theta) = \bigcup_{i=0}^{N} \{f(\theta)\}$ and then consider a modified objective function that uses the expected value of the sample distribution: $\Phi_E(\theta) = \overline{x_\theta}$, where $\overline{x_\theta} = \frac{1}{N} \cdot \sum_{x_i \in Ls(\theta)} x_i$. Then, Φ_E can be used directly instead of f as the global objective function.

6.4 Adaptive Penalty term

In order to deal with the variability of the distributions, (Painton and Diwekar, 1995) propose to add to the expression of Φ_E, a penalty factor based on the scaled standard deviation $\sigma_\theta = \sqrt{\frac{\sum_{x_i \in Ls(\theta)} (x_i - \overline{x_\theta})^2}{N}}$, weighted by a quantity that depends on the temperature. More specifically, their expression for Φ_E is:

$$\Phi_E(\theta) = \overline{x_\theta} + b(T) \cdot \frac{2 \cdot \sigma_\theta}{\sqrt{N}} \tag{11}$$

$$b(T) = \frac{b_0}{k^T} \tag{12}$$

Where b_0 is a small constant, k is a constant that represents the rate of increase of $b(T)$ and T the temperature of the system.

Given that at the beginning, the temperature is *high*, the search is wide into the search space. Thus, accepting non-significant changes has little adverse effect, although it leads to less evaluation of the objective function (which is costly to compute in principle).

6.5 Significance testing

In this article, we take a slightly different approach, by integrating a standard statistical test directly in the metropolis criterion, and by using Φ_E as only the expected value. Not only are such tests widely available in existing libraries, additionally, we avoid the hassle of adding two more constants to the estimation algorithm.

Depending on the properties of the distribution, different tests may be applied. The most widely used test for significance is the unpaired two-sample student t-test. However three important pre-requisites must be met before the test can be applied.

The distributions of samples in the groups must be normal, the samples in the groups must be independent, and the variances of the groups must be equal. In the case of simulated annealing under uncertainty, there is no guarantee that the distributions all keep these properties. Thus, in order to use the t-test (Press et al., 1988), ot would be required to test the hypotheses for every new re-sampling of the objective function. This could be achieved by applying the Shapiro–Wilk (Shapiro and Wilk, 1965) test to test the normality assumption and Levene's test (Levene, 1960) for the equality of variances.

In case the equality of variances criterion is not met, there is an alternative, Welch's t-test (Welch, 1947) that does not make that assumption. However, if the distribution is not normal, there is no other choice but to apply a non-parametric test.

A much simpler alternative, is to directly apply a non-parametric significance test, such as the Mann–Whitney U unpaired test, as there are no assumption about the distribution of the samples. The only requirement is that the values are ordinal and not regularly paced.

6.5.1 Modified Metropolis criterion

Since the objective function returns a numerical value, the Mann–Whitney U can be applied directly in the metropolis criterion as such:

$$P(\Delta f) = \begin{cases} 1 & \text{if } \Delta Phi_E < 0 \text{ and } p_\theta < \alpha \\ e^{\frac{-\Delta Phi_E}{T}} & \text{otherwise} \end{cases} \tag{13}$$

The test is based on the formulation of a null-hypothesis that states that the distribution of the two groups of samples are equal. Here, p_θ is the p-value (probability of true positives) resulting from the test and α a

threshold. If the p-value is below the threshold, then we reject the null-hypothesis and conclude that the distributions must be different.

For example with $\alpha = 0.05$ the null-hypothesis will be rejected if the probability of having a type I error is more that 95%: in other words, $(1-p) > (1-\alpha)$.

The Mann–Whitney U (Mann and Whitney, 1947) is based on the comparison of the ranks of the samples within each group. The first step toward calculating the p-value is to group all the sample together, to sort them in ascending order, and to assign a rank to each value. From the respective ranks of the samples, can be computed R_θ, is the sum of ranks for $Ls(\theta)$ and $R_{\theta'}$ the sum of ranks for $Ls(\theta^{'})$. Here, the number of samples for both groups are always equal to N.

6.5.2 Computing the p-value

From the sum of ranks, the U statistic can be computed as $U = min\left(R_\theta - \frac{N(N-1)}{2}; R_{\theta'} - \frac{N(N-1)}{2}\right) = min\left(R_\theta; R_{\vec{\theta'}}\right) - \frac{N(N-1)}{2}$.

For a sufficiently large N, the U statistic is an approximation of the normal distribution, and after applying a standardization by calculating the z statistic, the p-value can be retrieved from the probability density function of the normal distribution:

$$z = \frac{U - \overline{U}}{\sigma_U} \qquad \overline{U} = \frac{N^2}{2} \qquad \sigma_U = \sqrt{\frac{N^2(2N+1)}{12}} \qquad p_\theta = N(z, \overline{U}, \sigma_U) = \frac{1}{\sqrt{2\pi}\cdot\sigma_U} e^{-\frac{1}{2}\left(\frac{z-\overline{U}}{\sigma_U}\right)^2} \quad (14)$$

Despite the apparent complexity of the process of calculating and testing the p-value, the Mann–Whitney U test is rather standard and available in many libraries for various programming languages, including Java that we used for our implementation.

7 Experimental protocol

The principle used for the evaluation of the variant of simulated annealing considered for the estimation of parameters in an uncertain setting, was to consider the Semeval 2007 Task 7 annotated corpus. We split the corpus into a training and a test set in order to evaluate the parameter search with the Ant Colony Algorithm for WSD originally proposed in (Schwab and Guillaume, 2011) and then further improved upon in (Schwab et al., 2012). Furthermore, we want to determine exactly how much data was necessary to obtain a good set of parameter values.

Normally, in order to obtain a reliable training and thus parameters, it would be necessary to have a somewhat larger test set and to split it into several parts by randomly aggregating words to perform a $k-fold$ cross validation. However, the Ant Colony algorithm is meant to work on a full text as it exploits its structure. Notably, the order of the words and the continuity of sentences are very important. Making cross-validation sets randomly would thus only create a very noisy training data. Even just picking sentences at random still compromises the global coherence of the solutions found, and thus the parameters.

This is why we only considered a training on successively larger portions of our training corpus. More specifically, the experiments were carried out with 6,12,18,24,30 and 36 sentences in order from the first sentences to the full text in multiples of 6.

Furthermore, it is apparent that the nature of the training text itself has a big importance on the resulting parameters found. It is necessary to use a text that is as general as possible in terms of the senses it uses and of its theme as well as lexically varied. Even though, $k-fold$ cross validation cannot be applied directly in our experimental setting, we have decided to perform the experiment by taking, in turn, each text as a training corpus, with the remainder as a test corpus. Thus, we may also study the effect of the text itself and its characteristics (type of text, average polysemy, etc).

The implementation of this *uncertain* simulated annealing, runs the ant colony algorithm and passes the results through to the scorer to obtain the F-score values. For the stochastic loop, for every new candidate configuration, the ant colony algorithm is systematically run 50 times in order to guarantee a sufficient level of statistical significance. The search was left to converge for each successive number of sentences and then, each resulting parameter values were tested over 100 executions of the ant colony algorithm on the test corpus in order to ascertain the quality of the parameters.

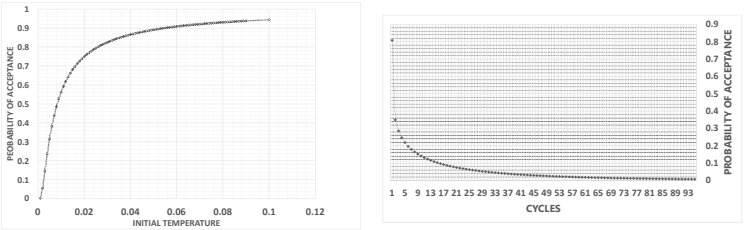
The results for each number of sentences are evaluated with relation to both classical lesk baseline and the baseline obtained with the parameters before the estimation (BL). We used standard statistical tests to guarantee the statistical significance of the comparisons.

8 Experiments and Results

8.1 Initial parameter for the simulated annealing estimator

Before starting the experiment, it is important to select the parameters of the simulated annealing-based algorithm we are using. In total there are 4 parameters. k_{stb} the number of cycles with no accepted new configurations before the system is considered to have converged. T_0 the initial temperature, and m and n, the parameters of the cooling schedule. We are aware that the manual estimation of the parameters for the simulated annealing algorithm contradicts the general orientation of this paper, however one of the main reason simulated annealing was chosen is because its behaviour is well known and the parameters can be set heuristically quite effectively. Indeed a vast body of work covers the algorithm and can be used to guide the parameter estimation. The main idea is to use SA on algorithms with more complex and/or not so well understood parameter models. In fact, this process could be consider as a form of bootstrapping of sorts.

For the number of cycles before convergence, 20 is a good compromise between efficiency and optimality. As for n and m, we wanted to have a cooling schedule with the highest possible probability for subsequent iteration, while approaching 0 after 100 iterations. To this effect, after various experiments, we selected the values $n = -3$ and $m = 1$.



(a) Choice of the initial temperature (b) Cooling schedule acceptance probabilities

Figure 2: Classical optimisation model versus stochastic optimisation model

However, the most important parameter is the initial temperature, as it needs to be set to a certain level of probability of acceptance. Of course, since the probability of acceptance depends of the difference between successive scores, the first step toward selecting the initial temperature is to measure the average difference between successive scores. Therefore, it is necessary to run the algorithm while only sampling parameter configuration, without making any decisions about acceptance or convergence. Thus, we ran the algorithm this way and obtained an average F-score delta of **0.57%** with a standard deviation of 0.0047.

It is described in the literature that a good value for the initial acceptance probability is 0.8. Thus, to

find the initial temperature, we can plot the graph of acceptance probability given the initial temperature (Figure 2a) by computing $e^{\frac{-0.58}{T_i}}$ with various values of T_i. Given that T is in the same unit as the objective function, the value must be situated between 0 and 1. In the present case, the value of T_i for which $e^{\frac{-0.58}{T_i}} = 0.8$ is **0.27**. After the parameters have been set, it is possible to directly plot the cooling schedule in terms of the average acceptance probabilities (Figure 2b) by using Equation 6.

8.2 Analysis

For each text used as a training set, we applied the standard Shapiro-Wilks (Shapiro and Wilk, 1965) non-normality test and the Levene's homoscedacity test (Levene, 1960) at a $\alpha = 0.1$ threshold, in order to verify the assumptions necessary to apply parametric statistical tests. For the results for Texts 1,2,3 the distributions resulting from each sentence led a to non-significant Shapiro Wilks, thus not allowing to reject the null hypothesis that the distributions are normal. Levene's test was significant for all texts, meaning that the variances are homogeneous. However for texts 4 and 5, the Shapiro-Wilks test was significant, thus making us reject the null hypothesis that the distributions are normal.

Thus, for text 1,2 and 3 we applied a standard Anova test and found a p-value that is also significant at the $\alpha = 0.01$ level. Furthermore we carried out the Tukey pairwise test, which significant differences in mean between most groups unless otherwise specified.

For texts 4 and 5 we applied the non-parametric rank-based Kruskal-Wallis test (**?**), which is similar to ANOVA for situations where the distributions involved are not normal. For the non parametric pair-wise test, we applied a non-parametric variant of Tukey's test, which the null hypothesis that "The probability of a sample from b being greater than a sample from a is superior to 0.5".

Table 3 and Figure 3a present the results for the first text as a training set in terms of value and in a graphical box plot representation.

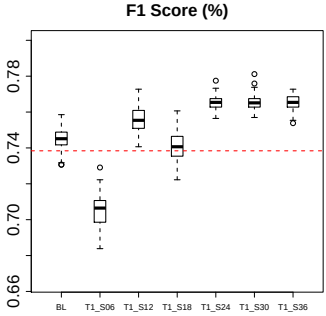
We can see for the first text in Table 3 and Figure 3a that with only 6 sentences, the results obtained after the estimations are worse that with the original parameters by 4.01%. With 12 and 18 sentences, the obtained results start to at the level of the *Before* (BL on the graph) baseline. For 12 there is an improvement of 1.04% and for 18, a decrease of 0.42%. It appears that adding sentences is not necessarily positive, given that from 12 to 18 the results quality decreased.

Then, the results for 24, 30 and 36 are notably better than the *Before baseline* by 1.98%, 1.98% and 1.99% and close, yet still below to the First Sense baseline with 1.07% to 0.8%. However, after 24 sentences, there aren't any significant improvements by adding more sentences.

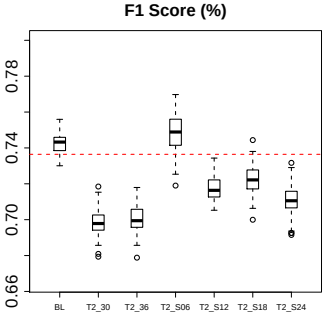
Sentences	$\overline{F_s}$(%)	σ_{F_s}	Not Sign.	E_a	E_{max}	δ_ϕ	E_0	ω	L_v	δ_v
First Sense	77.59									
Before	74.54	0.54	$\emptyset$	1	8	0.900	20	10	90	0.900
6	70.51	0.86	$\emptyset$	1	2	0.650	10	5	90	0.360
12	75.58	0.69	$\emptyset$	12	8	0.101	19	24	90	0.870
18	74.12	0.81	$\emptyset$	7	9	0.457	20	19	78	0.900
24	76.52	0.38	30, 36	18	45	0.922	26	10	60	0.979
30	76.52	0.41	24, 36	11	20	0.903	11	8	40	0.927
36	76.53	0.38	24, 30	17	60	0.490	20	10	92	0.359

Table 3: Results in terms of the average F_{score} and its standard deviation σ_{F_s} for a given set of parameter values. Unless specified, the pairwise Tukey HSD test is significant with $\alpha = 0.01$.

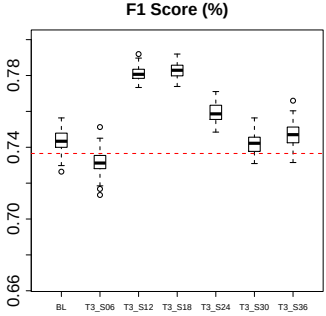
As for the values of the parameters, only a few conjectures can be drawn. It appears first that the value of some parameters (L_v) have very little effect on the quality of the results, while others (E_a, E_max, E_0), have



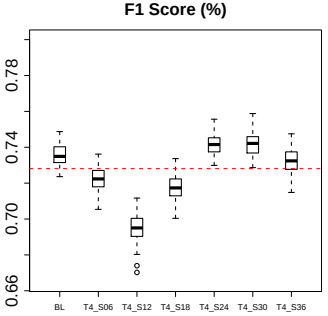
(a) Training on Text 1, Test on Texts 2,3,4,5



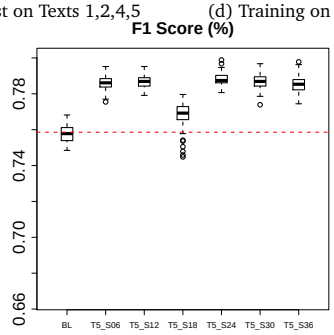
(b) Training on Text 2, Test on Texts 1,3,4,5



(c) Training on Text 3, Test on Texts 1,2,4,5



(d) Training on Text 4, Test on Texts 1,2,3,5



(e) Training on Text 5, Test on Texts 1,2,3,4

Figure 3: Box plots of the results for the training on Text 1 with different numbers of sentences compared to the Lesk algorithm 3(dashed line) and the Baseline of the parameters before the estimation.

Sentences	$\overline{F_s}(\%)$	σ_{F_s}	Not Sign.
Lesk	73.64		
Before	74.25	0.53	$\emptyset$
6	74.86	0.74	$\emptyset$
12	71.74	1.02	$\emptyset$
18	72.27	0.65	$\emptyset$
24	71.07	0.76	$\emptyset$
30	69.81	0.53	36
36	70.06	0.70	30

(a) Results for Text 2 as a training corpus

Sentences	$\overline{F_s}(\%)$	σ_{F_s}	Not Sign.
Lesk	73.65		
Before	74.54	0.57	30
6	73.15	0.64	$\emptyset$
12	78.10	0.37	18
18	78.25	0.38	12
24	78.89	0.53	$\emptyset$
30	74.18	0.54	BL
36	74.67	0.68	$\emptyset$

(b) Results for Text 3 as a training corpus

Sentences	$\overline{F_s}(\%)$	σ_{F_s}	Not Sign.
Lesk	72.80		
Before	73.57	0.57	$\emptyset$
6	72.18	0.73	$\emptyset$
12	69.46	0.79	$\emptyset$
18	71.75	0.74	$\emptyset$
24	74.16	0.59	30
30	74.19	0.66	24
36	73.22	0.69	$\emptyset$

(c) Results for Text 4 as a training corpus

Sentences	$\overline{F_s}(\%)$	σ_{F_s}	Not Sign.
Lesk	75.85		
Before	75.77	0.52	$\emptyset$
6	76.60	0.37	12,24,30,36
12	78.68	0.33	24,30
18	76.86	0.69	$\emptyset$
24	78.79	0.34	6,12,30
30	78.66	0.37	6,12,24,36
36	78.53	0.44	6,30

(d) Results for Text 5 as a training corpus

Table 4: Results in terms of the average F_{score} and its standard deviation σ_{F_s} for the parameters found with each number of sentences of training data.

a profound effect. We can observe that for all three number of sentences 24, 30, 36, we systematically have: $E_a < E_{max}$, $E_a \leq E_0$ and finally $\delta_\phi \simeq \delta_\nu$. Given that for the number of sentences 24, 30 and 36 the distribution are almost equal, we can hypothesise that there are to some degrees equivalences between some parameter value combinations.

We are also interested in the effect of the number of sentences by using the other texts of the corpus as training data. Furthermore we want to evaluate the fitness of certain types of texts for the purposes of parameter estimation for WSD. Tables 4a, 4b, 4c and 4d present the results for Texts 2,3,4,5 respectively as training sets. We will not, for these texts do a detailed analysis of the parameter values found, but will only look at the general trends with relation to the number of sentences and the properties of the texts. Figures 3b,3c,3d and 3e present box plots of the resulting score distributions for each text.

It is apparent that the quality of the parameter estimation widely depends on the nature of the texts used, as well as the quantity of data used for the training. For text 2, the results with any more than 6 sentences are all below the Lesk baseline, which could be indicative of a few general introductory sentences followed by very specific vocabulary. Text two is in fact an extract from the Wall Street Journal summarising a public scandal using very specific terms.

There are other cases, such as for text 3, where over fitting occurs very rapidly as the number of sentences increases. Such a behaviour could be explained by latter sentences of the text being domain specific and detrimental to the generality of the training. We observe a similar behaviour with text 4 towards the middle of the text. Text 3 is an extract from the wall street journal and is about a journalism convention, whereas Text 4 is a Wikipedia article about computer programming.

Text 5, a literary text extracted from a book, features the most steady training material, Indeed, except with 18 sentences, the resulting distributions from the estimated parameters are roughly equivalent as

indicated by the non-significant differences. The results are about at the same distance above the Lesk baseline than in Text 1 with 24 sentences and above. Such a text would be very desirable for parameter training, as even little data is enough to obtain a very general training.

While it is not possible to make use of cross validation by taking random sentences, it may be a future possibility to take chunks of texts containing several sentences unified thematically or semantically for example and that have good training properties. Thus making a more formal cross-validation possible and allow to find the best combination of sentences for the purposes of parameter estimation with more systematic criteria.

9 Conclusions and Perspectives

We have presented and evaluated a variant of simulated annealing for uncertain cost functions. More specifically, we have adapted the work of (Painton and Diwekar, 1995) to use standard statistical tests, as well as integrated some elements of adaptivity inspired by (Ingber, 1996)'s Adaptive Simulated Annealing. We performed experiments on Semeval 2007 task 7 and found that the quality of the results from a given set of estimated parameters very much depend on the nature of the text. However, from the texts in our corpus it appears that very general texts with a diverse yet non domain specific lexicon are the best choice for the estimation of parameters. As exhibited for one of the texts, even very small training sets are sufficient to obtain good results granted the text has the right properties.

For future work concerning this parameter estimation approach, we are planning some more thorough experiments involving the other texts, as well parts of SemCor. Furthermore, there are many potential ways to improve the parameter search algorithm, notably through the integration of more adaptive elements from Adaptive Simulated Annealing. Furthermore, it may be beneficial to evaluate a variant where the statistical test is integrated in the objective function as a penalty term, like (Painton and Diwekar, 1995) propose with the standard deviation.

Furthermore, it would be very interesting to adapt our approach to perform joint optimisation of features and parameters as done in (Daelemans et al., 2003), even though the stochastic aspect would make it a much lengthier process in order to ensure statistically valid results.

Resources and programs

The implementations of both the parameter estimation and the Ant Colony Algorithm are both available online as Free Software under the GNU Lesser General Public License: `https://forge.imag.fr/projects/formica/`. More information is available on the WSD page of our research group: `http://getalp.imag.fr/xwiki/bin/view/WSD/`.

Funding

This research was funded by the French Ministry Research Allocation and by the Traouiero ANR project.

References

Bolhasani (2004). Parameter estimation of vehicle handling model using genetic algorithm. 11(1-2):121–127.

Cowie, J., Guthrie, J., and Guthrie, L. (1992). Lexical disambiguation using simulated annealing. In *COLING '92*, pages 359–365, Nantes, France. ACL.

Daelemans, W., Hoste, V., Meulder, F. D., and Naudts, B. (2003). Combined optimization of feature selection and algorithm parameters in machine learning of language. In *In Proc of the 14th European Conference on Machine Learning, ECML 2003*, pages 84–95, Cavtat-Dubrovnik, Croatia.

Decadt, B., Hoste, V., Daelemans, W., and Van den Bosch, A. (2004). Gambl, genetic algorithm optimization of memory-based wsd. In Mihalcea, R. and Edmonds, P., editors, *Senseval-3: Third International*

Workshop on the Evaluation of Systems for the Semantic Analysis of Text, pages 108–112, Barcelona, Spain. Association for Computational Linguistics.

Gelbukh, A., Sidorov, G., and Han, S.-Y. (2003). Evolutionary approach to natural language Word Sense Disambiguation through global coherence optimization. *WSEAS Transactions on Communications*, 1(2):11–19.

Goutte, C. and Gaussier, E. (2005). A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *ECIR 27th European Conference on Information Retrieva*, Santiago de Compostela, Spain.

Ingber, L. (1996). Adaptive simulated annealing (asa): Lessons learned. *Control and Cybernetics*, 25(1):33–54.

Katare, S., Bhan, A., Caruthers, J. M., Delgass, W. N., and Venkatasubramanian, V. (2004). A hybrid genetic algorithm for efficient parameter estimation of large kinetic models. *Computers & Chemical Engineering*, 28(12):2569 – 2581.

Kay, S. M. (1993). *Fundamentals of statistical signal processing: estimation theory*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Kim, K.-J. and Diwekar, U. M. (2002). Efficient combinatorial optimization under uncertainty. 1. algorithmic development. 41(5):1276–1284.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Lee, Y. H., Park, S. K., and Chang, D.-E. (2006). Parameter estimation using the genetic algorithm and its impact on quantitative precipitation forecast. *Annales Geophysicae*, 24:3185–3189.

Levene, H. (1960). Robust tests for equality of variances. In *Ingram Olkin, Harold Hotelling, et alia.*, pages 278–292. Stanford University Press.

Mann, H. B. and Whitney, D. R. (1947). On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50–60.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Painton, L. and Diwekar, U. (1995). Stochastic annealing for synthesis under uncertainty. *European Journal of Operational Research*, 83(3):489 – 502.

Pan, Z., Chen, Y., Kang, L., and Zhang, Y. (1995). Parameter estimation by genetic algorithms for nonlinear regression. In *in G.Z.Liu (Ed.), Optimization Techniques and Applications, Proceedings of International Conference on Optimization Technique and Applications '95*, pages 946–953.

Paterakis, E., Petridis, V., and Kehagias, A. (1998). Genetic algorithm in parameter estimation of nonlinear dynamic systems. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, PPSN V, pages 1008–1017, London, UK, UK. Springer-Verlag.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1988). *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, New York, NY, USA.

Schwab, D., Goulian, J., Tchechmedjiev, A., and Blanchon, H. (2012). Ant colony algorithm for the unsupervised word sense disambiguation of texts: Comparison and evaluation. To be published.

Schwab, D. and Guillaume, N. (2011). A global ant colony algorithm for word sense disambiguation based on semantic relatedness. In *Highlights in Practical Applications of Agents and Multiagent Systems*, volume 89 of *Advances in Intelligent and Soft Computing*, pages 257–264. Springer Berlin / Heidelberg.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611.

Sharman, K. and McClurkin, G. (1989). Genetic algorithms for maximum likelihood parameter estimation. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*, pages 2716 –2719 vol.4.

Veronis, J. and Ide, N. M. (1990). Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *Proceedings of the 13th conference on Computational linguistics - Volume 2*, COLING '90, pages 389–394, Stroudsburg, PA, USA. Association for Computational Linguistics.

Welch, B. L. (1947). The Generalization of 'Student's' Problem when Several Different Population Variances are Involved. *Biometrika*, 34(1/2):28–35.

Yao, L. and Sethares, W. (1994). Nonlinear parameter estimation via the genetic algorithm. *Signal Processing, IEEE Transactions on*, 42(4):927 –935.