Noisy Sequence Classification with Smoothed Markov Chains

Pierre Dupont[1,2]

[1] Department of Computing Science and Engineering (INGI)
Université catholique de Louvain
Place Sainte Barbe, 2
B-1348 Louvain-la-Neuve - Belgium
Pierre.Dupont@uclouvain.be
http://www.info.ucl.ac.be/~pdupont/

[2] UCL Machine Learning Group
http://www.ucl.ac.be/mlg/

Abstract :

This paper is concerned with sequence classification using Markov chains when classification noise is included in the learning data. These models offer a direct generalization of a Multinomial Naive Bayes classifier by taking into account dependences between successive events up to a certain history length. Our study shows that smoothed Markov chains are very robust to classification noise. The relation between classification accuracy and test set perplexity, often used to measure prediction quality, is discussed. The influence of varying the model order is also studied from an experimental viewpoint. Experiments are conducted both on a gender classification task from spelling of first names and splicing region classification in DNA sequences. The first set of experiments also illustrate the superiority of smoothed Markov chains to classify noisy sequence over an automaton learning technique using boosting.

Keywords: Sequence Classification, Noisy Data, Markov chains, Smoothed N-grams.

1 Introduction

Markov chains are used in a wide variety of contexts such as biological sequence modeling (Durbin *et al.*, 1998), language modeling for speech recognition (Katz, 1987) or WEB pages indexing (Page *et al.*, 1998), to name a few. Markov chains are generative models of stochastic processes for which the dependence between successive events is assumed to be bounded according to the so-called Markov property. In other words, the prediction of future events is based solely on a fixed number of past events in the sequence. These notions were already used by Shannon for relating the definition of

To appear in CAp 2006, Conférence d'Apprentissage, Trégastel (France), *Presses Universitaires de Grenoble*, 2006.

entropy to the task of predicting English texts (Shannon, 1951). The fixed dependence in the past can be generalized to a variable history length (see, *e.g.*, (Kermorvant & Dupont, 2002; Begleiter *et al.*, 2004)) but in all cases a maximal model order, that is a maximal relevant history, is assumed.

When Markov chains, also known as N-grams, are used for prediction, the quality of the modeling is generally measured by computing the (log-)likelihood or, equivalently, the perplexity of previously unseen sequences. These values measure how well the model fits the underlying unknown process distribution being represented by independent test samples. Better models offer higher test likelihoods (smaller test perplexities).

In the present work we are interested in using Markov chains for sequence *classification* rather than *prediction*. At first glance, good predictive models should offer good classification performances. Indeed, according to Bayesian decision theory (Duda *et al.*, 2000), minimizing classification probability of error follows from maximizing posterior class probabilities. Maximum likelihood models are then optimal when uniform class priors can be assumed. There are nevertheless several questions to be addressed.

Practical estimation of Markov chains relies on smoothing techniques since the number of parameters to be estimated grows exponentially with the model order. Smoothing aims at attributing non zero probabilities to unseen events, which are (sub-)sequences never observed in the learning data. Smoothing is generally performed by discounting some probability mass from the observed events and by distributing this mass on the unseen events. Extensive experimental works have shown that smoothing increases prediction accuracy (Chen & Goodman, 1998). As smoothed models are no longer maximum likelihood models, it is worth investigating whether better smoothed models also offer better classification accuracy.

A related question is the choice of the optimal model order. Provided good smoothing techniques are used, higher order models typically predict better than standard (order 1) Markov chains. Second or third order models (3-grams or 4-grams) are typically the best predictors for language modeling tasks. If increasing the model order improves prediction accuracy, how does it affect classification performances?

Finally, real data is very often noisy. Noise can introduce various type of sequence distortions such as deletions, insertions or substitutions of individual symbols or subsequences. Classification noise replaces the class label of a given sequence by another class label. Classification noise is generally considered harder to deal with since a single class swap generally corresponds to several editing operations to recover a sequence from the original class. How does classification noise in the learning data affect classification accuracy of new sequences? Does the best Markov model selected in a noise free setting also performs best if classification noise is added in the learning data? In particular, how does noise influence the selection of the optimal model order?

We address in this paper the above questions from an experimental perspective. This is also motivated by the comparison with an alternative, and much more sophisticated, approach to noisy sequence classification (Sebban *et al.*, 2004). The authors proposed an automaton induction technique designed to deal with classification noise. To do so, they relied on a Markov chain model to constrain an automaton learning technique by state merging. The Markov model was used to evaluate the confidence of the class labels and this information was subsequently introduced in a boosting scheme. This

interesting work raises at least one question: what would be the classification result if the confidence oracle, that is the Markov model, was used alone? Comparative experimental results detailed in section 5 answer this question on a representative dataset.

The rest of this paper is structured as follows. Section 2 reviews briefly the definition of Markov chains and some related smoothing techniques. The computation of sequence likelihood based on the Markov assumption and its use as a decision rule for classification is detailed in section 3. This formulation illustrates that a Markov chain classifier forms a direct generalization of the multinomial Naive Bayes classifier. Section 4 describes the datasets used in our experiments. We present here several statistics including sequence length histograms and class overlaps. Section 5 summarizes the conclusions we can draw from our experiments.

2 Smoothed Markov Chains

Markov chains are models of stochastic processes for which the prediction of an event is based solely on a fixed number of past events. They are formally defined as follows.

Definition 1
A discrete time Markov Chain (MC) *is a stochastic process* $\{X_t | t \in \mathbb{N}\}$ *where the random variable* X *takes its value at any discrete time* t *in a countable set* W *and such that:*

$$P[X_t = w | X_{t-1}, X_{t-2}, \ldots, X_0] = P[X_t = w | X_{t-1}, \ldots, X_{t-p}].$$

This condition states that the probability of the next outcome only depends on the last p *values of the process (Markov property). When the set* W *is finite, the process forms a* p *order finite state MC.*

For sequence modeling, the set W is the finite alphabet from which the sequences are built. The relevant history $X_{t-1}, \ldots, X_{t-p}$ is simply denoted h when the model order is made implicit. Similarly, h_{-1} denotes the history restricted to the $p-1$ last symbols: $h_{-1} = X_{t-1}, \ldots, X_{t-(p-1)}$. The probability of a given symbol w in a sequence is thus computed as $P(w|h)$ where h is the relevant history considered. An *N-gram* simply denotes, in the language modeling literature, an $N-1$ order finite state Markov Chain and a 2-gram thus corresponds to a standard (order 1) Markov chain.

N-grams estimation from learning data is based on the counts of some subsequences. The count $C(h)$, respectively $C(h, w)$, refers to the number of times the subsequence h, respectively h followed by the symbol w, occurs in the learning data. The maximum likelihood estimation of $P(w|h)$ is given by

$$\hat{P}(w|h) = \begin{cases} \frac{C(h,w)}{C(h)} & \text{if } C(h) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

An N-gram built on the alphabet W has $|W|^N$ parameters which define all possible values $\hat{P}(w|h)$. Hence possible events will be assigned a zero probability when their associated counts are equal to 0. This is often observed, especially for high order models, even for very large datasets since the number of parameters grows exponentially

with the model order. Such a large number of parameters also results in poorly estimated probabilities even for seen events. The classical workaround relies on smoothing techniques to correct the maximum likelihood estimates.

2.1 Back-off smoothing

One of the most popular scheme used for language modeling is based on a so-called recursive *back-off* scheme (Katz, 1987). We recall below an improved back-off modeling proposed in (Kneser & Ney, 1995).

$$\hat{P}(w|h) = \begin{cases} \dfrac{C(h,w) - d_c}{C(h)} + \gamma(h)\hat{P}_{back}(w|h) & \text{if} \quad C(h,w) > 0, \quad (2) \\ \gamma(h)\hat{P}_{back}(w|h) & \text{if} \quad C(h,w) = 0 \text{ and } C(h) > 0, \quad (3) \\ \hat{P}_{back}(w|h) & \text{if} \quad C(h) = 0, \quad (4) \end{cases}$$

with

$$\gamma(h) = \sum_{w:C(h,w)>0} \frac{d_c}{C(h)}. \quad (5)$$

In equation (2), d_c denotes a discounted value subtracted from the counts of seen events ($C(h,w) > 0$). This discounted value may depend on the count $C(h,w)$ as in Turing-Good discounting (Katz, 1987), on the number of distinct seen events as in Witten-Bell discounting (Witten & Bell, 1991) or may be constant as in the case of absolute discounting (Ney *et al.*, 1994). The discounted probability mass $\gamma(h)$ is distributed to unseen events in proportion to their back-off estimates $\hat{P}_{back}(w|h)$ as defined in equation (3). The factor $\gamma(h)$ can also be considered as a normalization factor to guarantee that the smoothed estimates $\hat{P}(w|h)$ define a proper distribution: $\sum_w \hat{P}(w|h) = 1, \forall h$. Note that $\gamma(h)$ is well defined only if the corresponding history has been observed in the learning data ($C(h) > 0$). Otherwise, equation (4) applies and the back-off distribution is used in all cases.

In the simplest case, the back-off distribution is defined as $\hat{P}_{back}(w|h) = \hat{P}(w|h_{-1})$, where h_{-1} denotes a smaller history (typically a 2-gram history when h denotes a 3-gram history). In other words, the back-off distribution is given by the lower order estimate. The same scheme is applied recursively down to a unigram model. The recursive nature of this model implies that a N-gram smoothed in this way is actually a variable order (from $N-1$ to 0) Markov chain. The base case of the recursion is an order 0 Markov chain (no history is considered) $\hat{P}(w) = \frac{C(w)}{\sum_w C(w)}$, which is always strictly positive provided every symbol of the alphabet has been observed at least once in the learning data[1].

The original back-off scheme proposed in (Katz, 1987) only included the first term of equation (2) (and, consequently, a different normalization factor $\gamma'(h)$ to be used in equation (3)). Here the back-off distribution is used also for seen events (second term of

[1] Otherwise, an additional back-off to a uniform distribution $P(w) = \frac{1}{|W|}$ can be used.

equation (2)). Indeed the back-off distribution can generally be more reliably estimated as it is less specific and thus relies on more data. The resulting model is a mixture of Markov chains of various orders.

Kneser and Ney proposed an alternative back-off distribution which was shown to perform better in prediction (Kneser & Ney, 1995):

$$\hat{P}_{back}(w|h) = \frac{C(.,h_{-1},w)}{\sum_{w'} C(.,h_{-1},w')} \tag{6}$$

where

$$C(.,h_{-1},w) = \sum_{g:g=h_{-1},C(g,w)>0} 1.$$

Here, $C(.,h_{-1},w)$ corresponds to the number of different smaller histories h_{-1} where the word w has been observed ignoring the frequency of these events.

2.2 Underlying principles

The back-off scheme reviewed in section 2.1 was shown to offer significantly better predictive models (Chen & Goodman, 1998) than Markov chains smoothed by adding virtual counts (the *add-one* method in its simplest form), a method known as additive or Laplace smoothing which is popular in text categorization (Peng & Schuurmans, 2003). Interestingly, back-off smoothing is not only effective in practice but also follows from some theoretical principles.

The first requirement (although not specific to back-off models) is *symmetry*, which states that any two symbols having the same frequency (or count value c) in the learning sample must also have the same probability estimate $\hat{P}_c$. This principle of symmetry clusters the events into equivalence classes according to their respective counts. If n_c denotes the number of distinct events occurring c times[2] in the learning sample, the total number of events is given by $E = \sum_c cn_c$ and the maximum likelihood estimate is simply $\hat{P}_c = \frac{c}{E}$.

In this context, a cross-validation technique such as the leave-one-out method can be applied to estimate some unknown parameter, in particular the optimal discounting factor d_c (see equation (2)). By definition, the leave-one-out method removes iteratively each observed event from the learning data to form a training set containing $E-1$ samples and a single heldout sample. This process is repeated E times so that all E samples are used as heldout sample. For events appearing once in the original learning data, this procedure simulates unseen events automatically. Within this framework, the training part is used to define the count equivalence classes and the parameters can be estimated by maximum likelihood on the heldout samples. This approach was shown to produce exactly[3] the discounting factor defined by the Turing-Good estimate (Ney *et al.*, 1995):

$$d_c = c - (c+1)\frac{n_{c+1}}{n_c} \quad \text{with the corresponding estimate } \hat{P}_c = \frac{(c+1)}{E}\frac{n_{c+1}}{n_c}. \tag{7}$$

[2] n_c is often called a *frequency-of-frequency* or *count-of-count* value.

[3] apart from a normalization constant.

In this setting, the total probability mass assigned to unseen events is given by $\hat{P}_0 n_0 = \frac{n_1}{E}$, which indeed depends only on the number of events appearing once in the original data set.

The Turing-Good estimate defined in equation (7) has two limitations. Firstly, it assumes strictly positive values for the frequencies of frequencies n_c of all events considered. This is generally true in practice only for small c values and the original maximum likelihood estimate should then be used for larger count values, together with an appropriate renormalization of all estimates (Katz, 1987). Secondly, it does not necessarily satisfy the additional requirement of *monotony*: $\hat{P}_{c-1} \leq \hat{P}_c$.

The absolute-discounting model relies instead on a fixed discounting value $d_c \stackrel{\triangle}{=} d, \forall c$. The addition of the monotony constraint to the leave-one-out estimation of this model gives the following upper bound[4] d_* to the optimal discounting coefficient (Ney *et al.*, 1995):

$$d \leq d_* = \frac{n_1}{n_1 + 2n_2}. \tag{8}$$

In most practical cases $n_1 > 0$ and $n_2 > 0$, which guarantees that $0 < d_* < 1$.

The use of a lower order back-off distribution can be better understood if one considers the joint distribution $P(w, h)$. It can be rewritten as $P(w, h_{-1}, w_{-p})$, where the history h of length p is decomposed into a smaller history h_{-1} of $p-1$ symbols and a symbol w_{-p} occurring p steps before w. When the event (w, h) has not been observed, the joint distribution can be approximated by the product of the marginals $P(w, h_{-1})P(w_{-p})$. Equivalently, the conditional distribution $P(w|h)$ is then approximated by $P(w|h_{-1})$.

Finally, the modified back-off distribution given in equation (6) results from an approximated solution to the leave-one-out procedure when the additional *marginal constraint* is considered. The basic idea is to determine the back-off distribution such that the marginal distribution of the resulting joint distribution $\hat{P}(h, w|h_{-1})$ is identical to the given distribution $\hat{P}(w|h_{-1})$:

$$\hat{P}(w|h_{-1}) = \sum_g \hat{P}(g, w|h_{-1}).$$

3 Sequence likelihood and classification rule

Equation (9) defines the likelihood $P(x|M)$ of a sequence $x = x_1 \dots x_{|x|}$ according to a MC M.

$$P(x|M) = \prod_{i=1}^{|x|} P(x_i|h, M) \tag{9}$$

The actual history h used to compute this likelihood depends on the order of the Markov chain (see definition 1). In the above expression $P(x_i|h, M) = \hat{P}(w|h)$ where x_i, the i-th element of the sequence, corresponds to the symbol w and $\hat{P}(w|h)$ is estimated from learning data for the model M as detailed in section 2.

[4]The actual discounting factor d is generally set equal to this upper bound in practice.

The quality of a model to predict a set of previously unseen sequences X is often assessed with the per symbol log likelihood $LL = \frac{1}{||X||} \sum_{x \in X} \log_2 P(x|M)$, where $||X||$ denotes the sum of the sequence lengths. Test set perplexity PP is a related measure defined as $PP = 2^{-LL}$. The higher the log-likelihood, or the lower the perplexity, the better the distribution defined by M fits the actual distribution observed in the test sample X.

When Markov chains are used for sequence classification, a distinct model $\hat{P}(x|C_i)$ is built on each subset of the leaning data associated to a specific class i . According to Bayes decision theory, the class $\hat{C}$ assigned to a new sequence x is the one maximizing the posterior class probability:

$$\hat{C} = \underset{i}{\operatorname{argmax}} P(C_i)P(x|C_i). \tag{10}$$

When class priors can be assumed uniform, this decision rule reduces to maximizing the class likelihoods: $\hat{C} = \operatorname{argmax}_i P(x|C_i)$.

To sum up good predictive models assign high likelihoods to previously unseen sequences representative of the unknown process distribution while maximum likelihood decision rule is optimal for classification when uniform priors can be assumed.

Equation (9) shows that the use of Markov chains for classification is a direct generalization of a multinomial Naive Bayes classifier. Indeed, in the simplest case, an order 0 model is considered. This is equivalent to assigning the probability of a symbol x_i independently of its history. Each symbol in a sequence are then considered independently of the others and the sequence likelihood reduces to $\prod_{i=1}^{|x|} P(x_i)$. Markov chains thus form a generalization of a Naive Bayes classifier when longer histories are considered.

The same observation was already made in the context of text classification (Peng & Schuurmans, 2003). In this case, generalizing a Naive Bayes classifier by considering a non null history combined with appropriate smoothing techniques was shown to be effective. The present work confirms these results in the particular case when the sequences used to estimate the models include high classification noise.

4 Datasets

We describe in this section the datasets which were used to assess the performance of smoothed N-grams (variable order Markov chains) to classify sequences when high level of classification noise is added in the learning data. We selected two datasets coming from very different domains and with different characteristics detailed below.

The `WF` dataset contains a set of first names which we aim at classifying according to their gender. The `Splice` data sets contains fragments of DNA sequences which have to be classified either as exon/intron or intron/exon boundaries. Some sequences are repeated in those datasets, sometimes with distinct class labels. Since sequence repetitions and class overlaps generally affect classification results, we detail these features.

4.1 WF dataset

The WF dataset contains sequences of letters corresponding to male or female first names. The number of sequences in each class is summarized in table 1. There are 313 male sequences appearing once and 366 male sequences appearing twice for a total of 1045 male sequences. There is no repetition in the female class[5] .

	Male	Female	Total
Number of sequences	1045	842	1887
Number of distinct sequences	679	842	1521

Table 1: Class distribution in the WF dataset.

The WF sequences are drawn from an alphabet of 27 symbols (26 letters and the symbol[6] -). Figure 1 summarizes the histogram of sequence lengths for each class. The Male class, respectively the Female class, has an average sequence length of 6.5, respectively 7.2. This dataset thus contains relatively short sequences. Moreover, as all names of this dataset start with the same letter (A), the first letter of each sequence is useless for class discrimination.

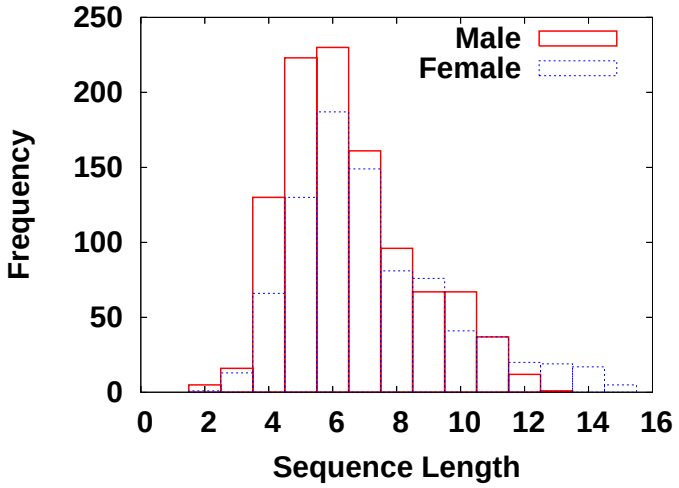

Figure 1: Histogram of sequence length in the WF dataset.

The original dataset was randomly split into 80% training and 20% test data[7]. Table 2 summarizes the number of sequences in each dataset. The classes overlap since

[5]We describe here the WF dataset containing 1887 sequences which was kindly provided to us by the authors of (Sebban *et al.*, 2004). We mention here certain peculiarities such as the unbalanced number of repetitions of this dataset. Even tough this unbalance is somewhat surprising, we did not edit the dataset to remove these repeated sequences in order to be able to compare their experimental results with ours.

[6]The symbol - is used in composed names such as ANNA-CHRISTINA.

[7]We use here a single training/test split as we follow the experimental protocol proposed in the boosting approach of (Sebban *et al.*, 2004). A classical 10-fold cross-validation would have been better but the computational cost of boosting was a limiting factor. Besides, we observed that our results were not significantly different when computed on different random folds of the same size as those reported here.

32 distinct sequences belong both to the `Male` and `Female` training sets while one sequence[8] is labeled both as `Male` and `Female` in the test set. The class overlap illustrates the natural ambiguity in gender classification based on first names. Additionally, classification noise is introduced by swapping a given percentage (from 5% up to 50%) of the class labels in the training data. The test labels are left unchanged.

	Dataset	Male	Female	Total
Number of sequences	Training	836	674	1510
Number of distinct sequences	Training	606	674	1280
Number of sequences	Test	209	168	377
Number of distinct sequences	Test	201	168	369

Table 2: Class distribution in the `WF` dataset.

4.2 Splice dataset

The `Splice` dataset is publicly available from the UCI Machine Learning Repository[9]. Splice junctions are points on a DNA sequence at which 'superfluous' DNA is removed during the process of protein creation in eukaryotes. The problem posed in this dataset is to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns (the parts of the DNA sequence that are spliced out). This problem consists of two subtasks: recognizing exon/intron boundaries (referred to as EI sites), and recognizing intron/exon boundaries (IE sites). In the biological community, IE borders are referred to as "acceptors" while EI borders are referred to as "donors".

More specifically, given a position in the middle of a window of 60 DNA sequence elements (called "nucleotides" or "base-pairs"), one has to decide whether this is a "intron $\Rightarrow$ exon" boundary (IE), a "exon $\Rightarrow$ intron" boundary (EI), or neither of them (N). Since we restrict here our attention to binary classification problems we used the subset of 1535 sequences labeled either as EI or IE.

Each data instance is characterized by 61 attributes apart from its class label. The first attribute is the instance name which we do not use in our experiments. The remaining 60 attributes denote the DNA nucleotide found at a given position in the window. In contrast with the `WF` dataset, the sequences are significantly longer and they all have the same length. The alphabet is composed of 8 letters[10].

Table 3 summarizes the number of sequences in each class together with their frequency of occurrence. This dataset contains 1352 distinct sequences out of which 123 sequences appear more than once. There is no class overlap in this dataset.

[8] The ambiguous test sequence is `AISSA`.

[9] The original data is available from `ftp://ftp.ics.uci.edu/pub/machine-learning-databases/molecular-biology/splice-junction-gene-sequences/`

[10] `A, T, C, G` stands for the specific nucleotides and 4 additional characters are used to indicate ambiguity: `D={A,G,T}`; `N={A,G,C,T}`; `S={C,G}`; `R={A,G}`.

Frequency	Donor (EI)	Acceptor (IE)	Total
1	619	610	1229
2	39	40	79
3	14	15	29
4	7	7	14
5	0	1	1

Table 3: Number of distinct sequences occurring with a certain frequency in the `Slice` dataset.

The original dataset was randomly split into 80% training (1228 sequences) and 20% (307) test data. Due to the high number of repeated sequences, 54 sequences were found both in the training and the test sets. These sequences were removed from the test set to guarantee no overlap between training and test conditions. The class distribution of the resulting 1481 sequences used in our experiments is summarized in table 4. Additionally, classification noise is introduced by swapping a given percentage (from 5% up to 50%) of the class labels in the training data. The test labels are left unchanged.

	Dataset	Donor	Acceptor	Total
Number of sequences	Training	614	614	1228
Number of sequences	Test	126	127	253

Table 4: Class distribution in the `Splice` dataset.

5 Experimental results

This section presents the experimental results obtained on both datasets. They illustrate the high resistance to classification noise offered by smoothed N-grams.

5.1 Gender classification from names (`WF` data).

Figure 2 illustrates the classification results obtained on the training sequences of the `WF` dataset when classification noise is introduced. Each plot reports the sequence perplexity attributed by a smoothed 2-gram for the `Male` class (x-axis) and the `Female` class (y-axis). Class priors are assumed equal here, which is a reasonable assumption for Male/Female classification[11]. The decision rule, represented by the dashed line in the figures, strictly corresponds to a minimum perplexity classifier, that is a maximum (per symbol) likelihood classifier. The introduction of 20 % classification noise in the training data results logically in more similar likelihoods assigned by both models (the points are overall closer to the dashed line). However, the classification accuracy does not degrade much: 85.76% (with 0% noise) versus 83.18% (with 20 % noise). In other

[11] Alternatively, class priors could have been estimated from the learning sample.

words, even tough the prediction of the two classes are closer to each other, the number of points on the wrong side of the decision boundary does not increase much. Interestingly, the same conclusion can be drawn from figure 3 reporting classification results on the test data. Here again, the introduction of 20% classification noise does not affect much the classification accuracy (going from 81.96% down to 79.31%).

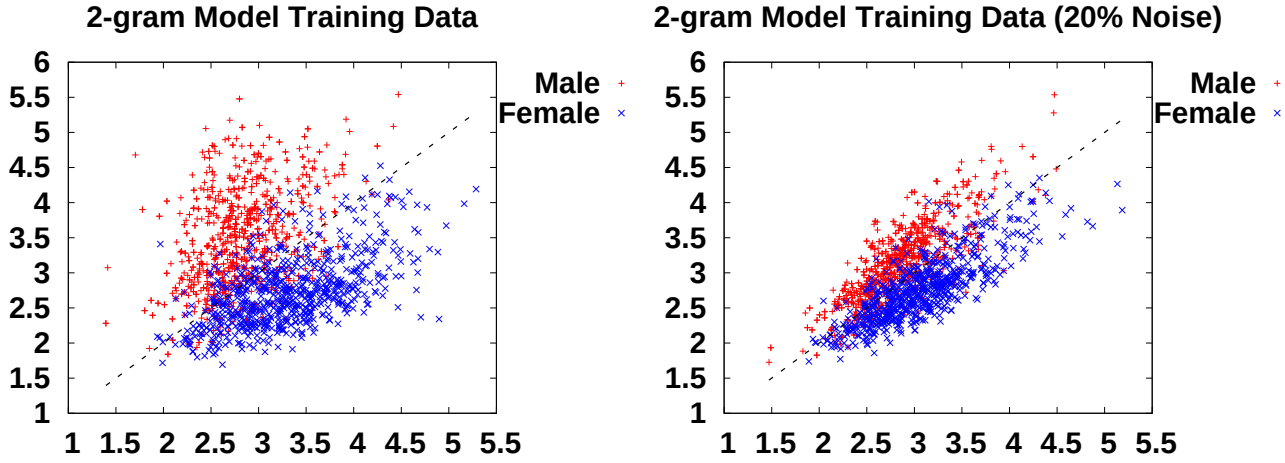

Figure 2: Training classification results on the `WF` datasets with 0% and 20% classification noise in the training data.

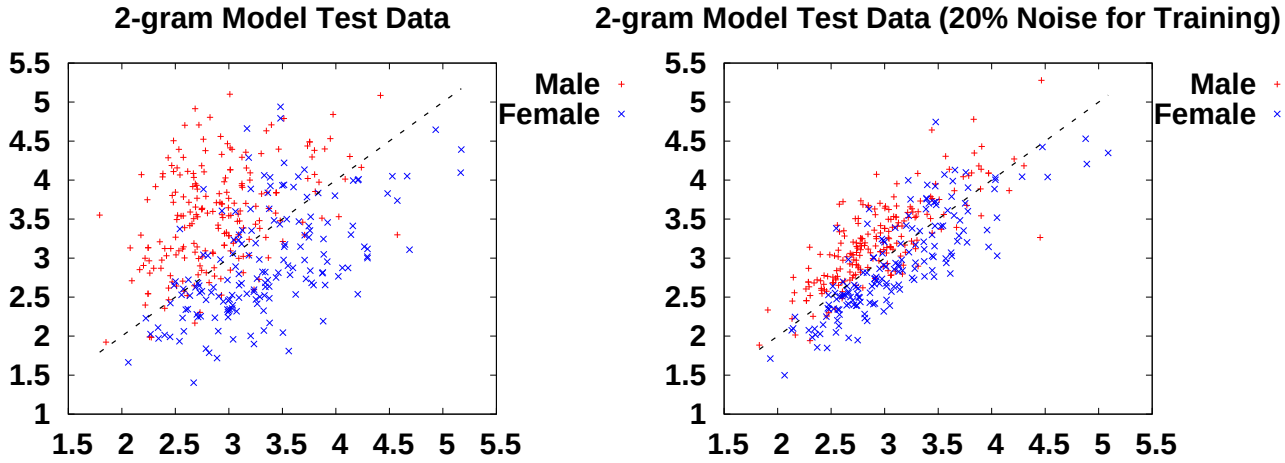

Figure 3: Test classification results of smoothed 2-gram models on the `WF` datasets with 0% and 20% classification noise in the training data.

Table 5 gives a more detailed analysis of the robustness to noise of these models. On the left, the classification confusion matrices are reported on the test data, respectively with 0% and 20% classification noise in the training. The columns indicate the true class labels while the rows indicate the predicted class based on the highest model likelihood. On the right, the global perplexities assigned to the same sequences by each model are reported. It can be observed that the addition of classification noise results in a slight increase of the perplexities of correctly classified sequences. The perplexity of incorrectly classified sequences is reduced showing that the construction of the models is influenced by examples of the wrong class. However, the perplexity remains higher for incorrectly classified examples and the confusion matrix is not drastically modified.

	Male	Female
Male (predicted)	179	38
Female (predicted)	30	130

	Male	Female
Male (predicted)	7.4	10.1
Female (predicted)	12.0	7.3

	Male	Female
Male (predicted)	172	41
Female (predicted)	37	127

	Male	Female
Male (predicted)	7.6	8.5
Female (predicted)	9.2	7.4

Table 5: Confusion matrices on the `WF` test data (left) and associated test perplexities (right). Results above, respectively below, refer to 0% classification noise, respectively 20% classification noise in the training data.

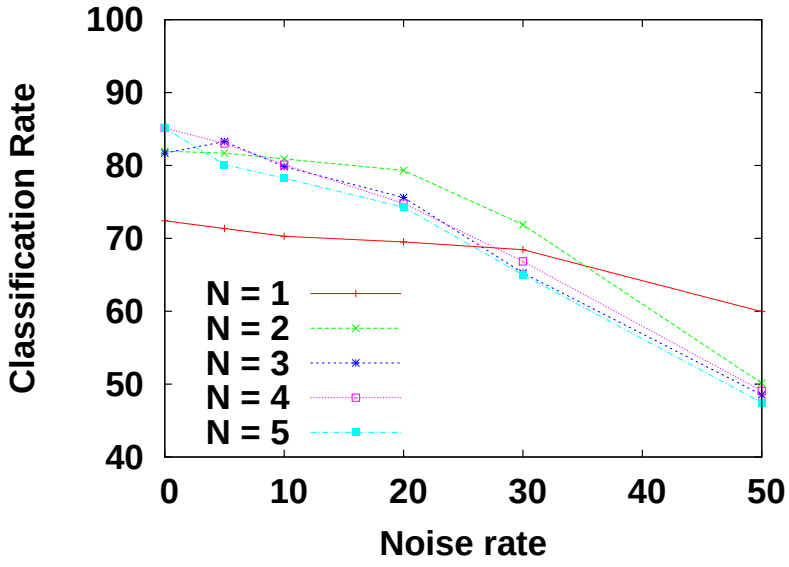

Figure 4: Classification results on the `WF` test data for various model orders and classification noise rates in the training data.

Figure 4 summarizes the results obtained with several noise rate (from 0% up to 50%) while varying the model order. 1-grams are equivalent to a multinomial Naive Bayes classifier (see section 3). Increasing the model order of well smoothed N-grams does improve the performance. On this dataset a smoothed 2-grams offers the best overall performance at various noise levels. Classification accuracy is fairly stable up to 20% noise and still above 70% with 30% classification noise in the training data. We attribute the relatively good performance[12] of a 1-gram model even with 50% noise as a consequence of the presence of repeated `Male` sequences. Hence classification into the `Male` class is more likely both in the training and test data. This data artefact does not compensate for the classification noise when higher order models are used.

Table 6 reports comparative[13] results on the `WF` dataset of a 2-gram classifier and the

[12] One would have expected roughly a 50% classification rate with 50% classification noise in the training data.

[13] We use here the same dataset and, to the best of our knowledge, we follow strictly the same experimental

automaton boosting approach proposed in (Sebban *et al.*, 2004). Classification error rates are reported here. The 2-gram was only used in a preprocessing phase as a confidence oracle in this alternative approach referred here as `BOOST`. Our results show that a 2-gram alone is significantly more robust to classification noise.

Noise rate	2-gram	BOOST
5%	18.3%	21.4%
10%	19.1%	22.7%
20%	20.7%	31.7%
30%	28.1%	?

Table 6: Classification error rates of a smoothed 2-gram and of the `BOOST` approach at various noise levels.

5.2 Splicing region classification in DNA sequences (`Splice` data)

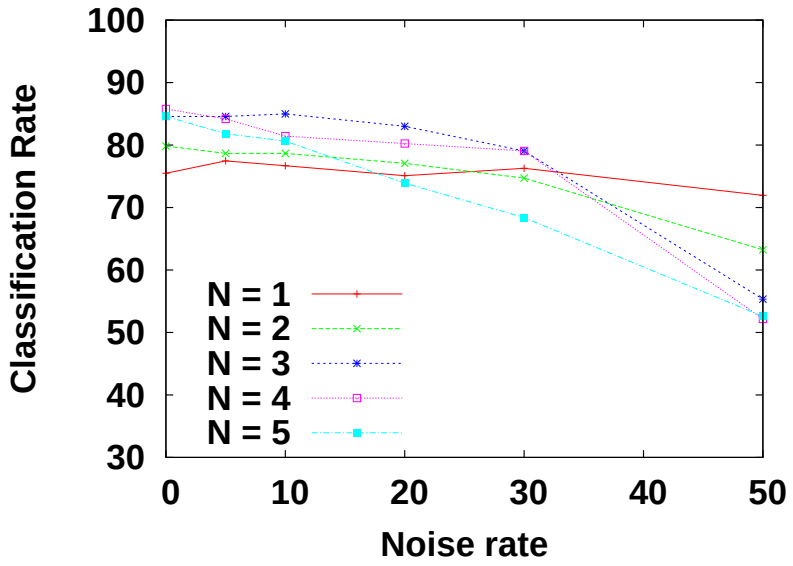

Figure 5: Classification results on the `Splice` test data for various model orders and classification noise rates in the training data.

Results similar to those presented in section 5.1 are obtained on the `Splice` dataset. Figure 5 summarizes the results obtained with several noise rate (from 0% up to 50%) while varying the model order. Increasing the model order of well smoothed N-grams does again improve performances. On this dataset, a smoothed 3-gram model offers the best overall performance at various noise levels. Classification accuracy is equal to 84.6 % with a 3-gram model and no noise and only goes down to 79.05 % with 30% classification noise included in the training data.

protocol as the one proposed in (Sebban *et al.*, 2004). We did not have access however to their actual split into training and test sets but we use the same proportions. We do believe that the better performance we observe especially with 20% classification noise are significant.

A 1-gram model correctly classifies 72 % of the test examples when 50% noise is introduced in the training data. We believe that this robustness is due to the repetitions of some sequences in the data, even if there is no overlap between training and test as explained in section 4.2. To confirm this hypothesis, all repetitions were removed both from training and test sets. The overall performance is reduced roughly by 10 %, for noise rate less or equal to 30%, as less data is available to estimate the models. Moreover all models have, as expected, roughly a 50% classification rate when 50% classification noise is introduced. This suggests that repeated sequences are indeed interesting to increase robustness to classification noise. Note that repetitions are legitimate in this data since the 123 distinct sequences appearing more than once form 306 occurrences. Among these 306 occurrences only 24 correspond to the same 12 instances repeated twice. The remaining 282 occurrences correspond to distinct instances but sharing with another instance the same fragment of DNA extracted. This redundancy is useful for classification even under noisy conditions.

6 Conclusion

We study in the present work sequence classification using Markov chains when classification noise is introduced in the learning data. Back-off smoothed models, also known as N-grams, are considered here. They define variable order Markov chains. These models were shown to be effective for language modeling tasks where there are used for prediction. Sequence classification is a distinct but related problem. The difference and similarities between them are discussed.

Experiments are conducted on a gender classification task from spelling of first names and splicing region classification in DNA sequences. They both illustrate the high robustness of smoothed Markov chains to classification noise. The results on the first task also show the superiority of these models over an automaton learning technique using boosting (Sebban *et al.*, 2004). In this alternative approach, Markov chains were also used but only as a preprocessing to estimate the confidence of the class labels. This information was subsequently introduced in a boosting scheme. Our experiments show that a well smoothed Markov chain alone performs significantly better and is more robust to noise.

The decision rule used here to classify unknown sequences is a minimal perplexity classifier or, equivalently, a maximum likelihood classifier. In this respect, Markov chains offer a natural generalization of a Multinomial Naive Bayes classifier by taking into account dependences between successive events up to a certain history length. The performance gain obtained with this generalization together with appropriate smoothing techniques had already been observed for text classification (Peng & Schuurmans, 2003). The present work confirms these results under high classification noise conditions in the learning data.

The proposed classifiers assign a single value to any new sequence. This value is directly related to the per-symbol log likelihood. A possible generalization would assign several values possibly to several sub-sequences of the original sequence. This would define a new feature space from which traditional classifiers could be built. We left this extension for our future work.

Acknowledgments

We would like to thank Marc Sebban for fruitful discussions about this work and for giving us access to the WF dataset.

References

BEGLEITER R., EL-YANIV R. & YONA G. (2004). On prediction using variable order markov models. *Journal of Arificial Intelligence Research*, **22**, 385–421.

CHEN S. & GOODMAN J. (1998). *An empirical study of smoothing techniques for language modeling*. Technical report, Computer Science Group, Harvard University. TR-10-98.

DUDA R., HART P. & STORK D. (2000). *Pattern Classification*. Wiley-Interscience, 2nd edition.

DURBIN R., EDDY S., KROGH A. & MITCHISON G. (1998). *Biological sequence analysis*. Cambridge University Press.

KATZ S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustic, Speech and Signal Processing*, **35**(3), 400–401.

KERMORVANT C. & DUPONT P. (2002). Improved smoothing for probabilistic suffix trees seen as variable order markov chains. In *Proceedings of the European Conference on Machine Learning*, p. 185–194, Helsinki, Finland.

KNESER R. & NEY H. (1995). Improved backing-off for m-gram language modeling. In *International Conference on Acoustic, Speech and Signal Processing*, p. 181–184, Detroit, Michigan.

NEY H., ESSEN U. & KNESER R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech and Language*, **8**, 1–38.

NEY H., ESSEN U. & KNESER R. (1995). On the estimation of 'small' probabilities by leaving-one-out. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(2), 1202–1212.

PAGE L., BRIN S., MOTWANI R. & WINOGRAD T. (1998). *The PageRank Citation Ranking: Bringing Order to the Web*. Technical report, Stanford Digital Library Technologies Project.

PENG F. & SCHUURMANS D. (2003). Combining naive bayes and n-gram language models for text classification. In F. SEBASTIANI, Ed., *Advances in Information Retrieval: Proceedings of The 25th European Conference on Information Retrieval Research (ECIR03)*, volume 2633 of *Lecture Notes in Computer Science*, p. 335–350: Springer-Verlag.

SEBBAN M., JANODET J., SUCHIER H. & NOCK R. (2004). Boosting grammatical inference with confidence oracles. In G. . SCHUURMANS, Ed., *International Conference on Machine Learning (ICML)*, p. 425–432.

SHANNON C. (1951). Prediction and entropy of printed English. *Bell System Technical Journal*, **30**, 50–64.

WITTEN I. & BELL T. (1991). The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression. *IEEE Transactions on Information Theory*, **37**(4), 1085–1094.