"© 2014 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works."

SEQUENTIAL LABELING WITH STRUCTURAL SVM UNDER THE F_1 LOSS

Guopeng Zhang and Massimo Piccardi

School of Computing and Communications, University of Technology, Sydney, Australia

ABSTRACT

Sequential labeling addresses the classification of sequential data and is of increasing importance for the classification and segmentation of video data. The model traditionally used for sequential labeling is the hidden Markov model where the sequence of class labels to be predicted is encoded as a Markov chain. In recent years, hidden Markov models and other structural models have benefited from minimum-loss training approaches which in many cases lead to greater classification accuracy. However, the loss functions available for training are restricted to decomposable cases such as the zero-one loss and the Hamming loss. Other useful losses such as the F_1 loss, equal error rates and others are not available for sequential labeling. For this reason, in this paper we propose a training algorithm that can cater for the F_1 loss and any other loss function based on the contingency table. Experimental results over the challenging TUM Kitchen Dataset depicting human actions in a kitchen scenario show that the proposed training approach leads to significant improvement of different performance metrics such as the classification accuracy (4.3 percentage points) and the F_1 measure (8.9 percentage

Index Terms— Sequential labeling, hidden Markov model, structural SVM, Hamming loss, F_1 loss.

1. INTRODUCTION

Sequential labeling has been drawing increasing attention from the image processing and machine learning communities [1, 2, 3, 4, 5]. Also known as tagging, or decoding, sequential labeling consists of the assignment of a sequence of class labels to a sequence of measurements. This problem is a fundamental problem in many image and video processing applications such as image segmentation, handwriting recognition, activity recognition, pose estimation, and also in fields as diverse as gene finding, protein secondary structure prediction, financial analysis and many others.

A simple, yet effective, model used for sequential labeling is the hidden Markov model (HMM) [6]. An HMM consists of two sets of variables: a sequence of class variables linked in a first-order Markov chain, and a sequence of measurements each depending only on the corresponding class variable. The parameters of this model have traditionally been learned by

maximising the likelihood function over a set of supervised examples. However, in recent years other styles of training have become increasingly popular; in particular, the extension of the support vector machine (SVM) to the classification of sequences [7, 8]. An SVM is trained by minimising the empirical loss over the set of examples instead of the likelihood function, typically leading to superior run-time performance. However, training requires the choice of a suitable loss function. Common loss functions include the misclassification rate, the (negative) precision at a fixed value of recall, the F_1 loss and several others [9]. Depending on the application, certain loss functions prove more suited than others to reflect a desirable performance. For instance, in the case of sparse detection, a balanced combination of precision and recall such as the F_1 loss is certainly more meaningful than the simple misclassification rate. For this reason, Joachims in [10] has provided algorithms for training an SVM under several, diverse loss functions. However, they do not cover the targeted case of sequential labeling. Therefore, in this paper we present an algorithm for training a sequential SVM using the F_1 loss as loss function. The experimental results over a challenging sequential action dataset (the TUM Kitchen Dataset depicting human actions in a kitchen setting [11]) show that the proposed method achieves higher performance than the state of the art in terms of both F_1 measure and classification accuracy.

2. SEQUENTIAL LABELING AND STRUCTURAL SVM

2.1. Sequential labeling

The formal goal of sequential labeling is to predict a sequence of class labels, $y=(y_1,\ldots,y_t,\ldots,y_T)$, for a given measurement sequence, $x=(x_1,\ldots,x_t,\ldots,x_T)$, where x_t is a feature vector at sequence position t and y_t is a corresponding discrete label, $y_t \in 1\ldots M$. In many cases, it is not restrictive to assume that y_t is a binary label (1: positive class; 0: negative class), obtaining multi-class classification as a combination of binary classifiers. Therefore, in the following we focus on this binary case. The most widespread model for sequential labeling is the hidden Markov model which is a probabilistic graphical model factorising the joint probabilities of labels and measurements as:

$$p(y,x) = p(y_1) \prod_{t=2}^{T} p(y_t|y_{t-1}) \prod_{t=1}^{T} p(x_t|y_t)$$
 (1)

The terms in the first product, $p(y_t|y_{t-1})$, are known as transition probabilities and the terms in the latter, $p(x_t|y_t)$, are known as emission densities. By restricting the emission densities to belong to the exponential family of distributions, i.e. $p(x_t|y_t) \propto exp(w_{em}^T f(x_t,y_t))$, and moving to a logarithmic scale, it is easy to see that the HMM joint log-probability is equivalent to the score a generalised linear system:

$$\ln p(y,x) \propto w^{T} \phi(y,x) = w_{init} f(y_{1}) + \sum_{t=2}^{T} w_{tran}^{T} f(y_{t}, y_{t-1}) + \sum_{t=1}^{T} w_{em}^{T} f(x_{t}, y_{t})$$
(2)

This generalised linear model is more suitable for discriminative training and notable cases are conditional random fields [12] and the structural SVM [8]. The inference problem consists of determining the best class sequence for a given measurement sequence:

$$\bar{y} = \underset{y}{\operatorname{argmax}} p(y, x) = \underset{y}{\operatorname{argmax}} (w^T \phi(y, x))$$
 (3)

This problem can be efficiently solved in O(T) time by the Viterbi algorithm working in logarithmic scale. Viterbi is an instance of dynamic programming where the sequence is scanned in a left-to-right manner and partial solutions accumulated. The final accumulated solution is guaranteed to be the global optimum [6].

2.2. Loss function

A loss function quantifies the loss, or cost, carried by a misclassification. If we assume that each sample carries a single class label, we can note the true label of sample i as y^i , our prediction as y, and the loss function as $\Delta(y^i,y)$. The simplest loss function is the zero-one loss, assigning a unit loss to every misclassification and a zero loss to every correct classification:

$$\Delta_{0-1}(y^i, y) = \begin{cases} 1 & \text{for } y \neq y^i \\ 0 & \text{for } y = y^i \end{cases}$$
 (4)

In this case, the average loss over a sample set is just equivalent to the misclassification, or error, rate. However, other loss functions are required for differentiating between different types of error. This is particularly important in sequential labeling, where a sequence of labels is predicted at once: we certainly wish to differentiate between major errors, where many labels may be predicted wrongly, and minor errors with only a few, incorrect labels. In this case, the most

common loss function is the Hamming loss that adds a loss fraction for every incorrect label [9]:

$$\Delta_{Hamming}(y^i, y) = \frac{1}{T} \sum_{t=1}^{T} y_t^i \oplus y_t$$
 (5)

where operator \oplus is the modulo-2 sum, or *XOR* boolean operator.

Although the Hamming loss is a more appropriate loss function than the zero-one loss for sequences as it accounts for gradual losses, it does not reflect desirable performance in some important cases. Let us consider a simple example: the ground truth, y^i , having 3 positive labels and 97 negative labels. In this case, a trivial classifier with constant negative output would carry a loss of only 3%. This asymmetric situation is very common in detection, where positive labels are generally less frequent than negative ones. A useful performance figure that accounts specifically for false positive and false negative errors is the F_1 measure:

$$F_1(y^i, y) = \frac{2 TP}{2 TP + FN + FP} \tag{6}$$

where TP is the number of correctly classified positive samples, FN the number of false negatives and FP the number of false positives. The F_1 measure is a figure of merit ranging between 0 (worst performance) and 1 (best) and it is then straightforward to define a loss function based on the F_1 measure as:

$$\Delta_{F_1}(y^i, y) = 1 - F_1(y^i, y) \tag{7}$$

In the case of the trivial classifier with constant negative output, the F_1 loss reaches its maximum value (100%) since TP is equal to zero. Other important loss functions are based on: a) the $\operatorname{precision}\left(TP/(TP+FP)\right)$ at a fixed value of $\operatorname{recall}\left(TP/(TP+FN)\right)$, b) the recall at a fixed value of precision, c) precision and recall at equal value, d) the F_β measure (a variant of the F_1 measure weighing FP and FN differently). All such losses can be similarly computed from the values of TP, FP, FN (the contingency table of prediction y versus ground truth y^i).

2.3. Structural SVM

SVM has been used widely in classification problems mainly thanks to its remarkable experimental accuracy [13]. The conventional SVM is a classifier for independent data, i.e. data where the {label, measurement} pairs are assumed independent of each other. However, the SVM has also been extended to the prediction of structured labels, i.e. multiple labels that have mutual dependencies in the form of sequences, trees and graphs and that possibly co-depend on multiple measurements [7, 8]. Given a set of N training instances $\{y^i, x^i\}$, $i = 1 \dots N$, structural (or structured-output) SVM finds the optimal model's parameters by constrained minimisation of the following objective function:

$$\sum_{i=1}^{N} \xi^{i} + \frac{1}{C} \|w\|^{2} \quad s.t.$$

$$w^{T} \phi(y^{i}, x^{i}) - w^{T} \phi(y, x^{i}) \ge \Delta(y^{i}, y) - \xi^{i},$$

$$\xi^{i} \ge 0, i = 1 \dots N, \forall y$$
(8)

The first term in the objective function $(\sum_{i=1}^N \xi^i)$ can be proven a convex upper bound for the loss over the training set, showing that the SVM is a minimum-empirical-loss classifier. The second term $(\|w\|^2)$ is a regulariser that disambiguates the scale of the w parameters. Parameter C is an arbitrary, positive coefficient that balances the two terms in the objective. In the constraints, function $w^T\phi(y,x)$ assigns a generalised linear score to a $\{y,x\}$ pair. In the case of sequential labeling, it is given by Eq. (2). Eventually, $\Delta(y^i,y)$ is the chosen loss function.

Due to the exponential number of values for y in Eq. (8), exhaustive verification of the constraints is infeasible. However, [8] has shown that it is possible to find ϵ -correct solutions in polynomial time by using only the "most violated" constraint for each sample, i.e. the class labels with the highest sum of score and loss:

$$\bar{y} = \underset{y}{\operatorname{argmax}} (w^T \phi(y, x^i) + \Delta(y^i, y)) \tag{9}$$

This problem is commonly referred to as "augmented inference". In the case of the zero-one loss or the Hamming loss, the augmented inference for sequences can be easily resolved by an appropriately weighted Viterbi algorithm. However, when the required loss is the F_1 , or any other non-decomposable loss functions, the Viterbi algorithm is not applicable anymore. An algorithm for these cases is the main contribution of this paper and is presented in the following section.

3. AUGMENTED INFERENCE FOR BINARY SEQUENTIAL LABELING UNDER THE F_1 LOSS

The solution of Eq. (9) is the crux of structural SVM for sequential labeling. Since the F_1 loss is not additive over the sequence's frames, substantial changes to conventional dynamic programming algorithms are required. We start by re-writing Eq. (7) as:

$$\Delta_{F_1}(y^i, y) = 1 - \frac{2(P - FN)}{2P + FP - FN} \tag{10}$$

Given any arbitrary ground-truth label sequence, y^i , the number of positive labels, P, in Eq. (10) is fixed. Therefore, the value of the F_1 loss only further depends on the number of false positives, FP, and false negatives, FN, in the predicted sequence, y. We present our strategy herewith: by making the prediction in left-to-right order along the sequence, the values of FP and FN can only increment or remain unchanged.

We can thus still approach the solution of Eq. (9) by dynamic programming by extending the state of a partial solution to include: a) the label of the current class, y_t , as in conventional Viterbi; b) the number of false positives, FP, in subsequence $y_{1:t}$; and c) the number of false negatives, FN, in sub-sequence $y_{1:t}$. We use notation $\psi(FP,FN,y_t)$ to indicate the $y_{1:t}$ sub-sequence with the highest score for the given extended state, and we note its score as $p(\psi)$. The generic induction step is as follows: at any time step, t, a partial solution is obtained by extending two of the partial solutions of time t-1 with the current prediction, y_t , and the corresponding increment of either FP or FN if the prediction is incorrect, or neither if correct. After the final time step, T, Eq. (9) is computed over all the stored sequences and the argmax returned. Algorithm 1 describes the solution formally.

During the computation of Algorithm 1, the number of stored sequences increases at every time step and reaches its maximum at the last time step, T. Given that FP and FN are bound by the number of negative and positive labels, N and P, respectively (with N+P=T), the number of stored sequences is upper bound by 2(P+1)(N+1). The final loop of Algorithm 1 runs over all values of FP, FN to find the maximum for Eq. (9): with obvious modifications, the final loop can be adapted to search for the maximum over fixed values of FP or FN, at equal rates FP=FN or, in general, any loss functions based on the contingency table.

4. EXPERIMENTAL RESULTS

The recognition of human activities in video is a major application of sequential labeling [14]. In this section, we evaluate our method on the TUM Kitchen Dataset which is a collection of activity sequences recorded in a kitchen environment equipped with multiple sensors [11]. In the kitchen environment, various human subjects were asked to set a table in different ways, performing actions such as Reaching, Carrying, TakingSomething and several others. The dataset contains multiple, simultaneous types of data: video data from 4 fixed overhead cameras, motion capture data extracted from the videos, RIFD tag readings from 3 fixed readers embedded in the environment, magnetic sensors readings from objects and the environment, and their annotated action labels. For our experiments, we have used the motion capture data (3-D body joint locations) which include 19 sequences for the left and right hands separately, each ranging in length between 1,000 and 6,000 frames. Each measurement is a 45-D vector.

We developed our software in MATLAB 2011 using the original SVM^{struct} from Thorsten Joachims and its MATLAB wrapper from Andrea Vedaldi [15, 16]. All experiments were performed on a PC with an Intel i7 2.4GHz CPU with 8 GB RAM. As data set, we divided the 19 left hand sequences into a training set with 6 sequences and a test set with the remaining 13, and we considered all the actions that had pos-

Table 1 : Comparison	of the average accuracy	y and average F_1 meas	sure on TUM kitchen data	set with different loss functions.

	Accuracy (%)			F_1 measure (%)		
Loss function	0-1 loss	Hamming loss	F_1 loss	0-1 loss	Hamming loss	F_1 loss
Reaching	20.02	61.39	52.68	9.83	29.21	28.30
TakingSomething	21.27	61.59	52.81	2.87	28.80	29.15
LoweringAnObject	48.51	49.43	62.90	52.18	38.53	53.95
ReleasingGrasp	22.91	49.31	60.30	12.14	21.37	27.30
Carrying	40.89	53.36	57.91	39.10	51.63	72.09
Average	30.72	52.96	57.28	25.04	33.53	42.44

Algorithm 1: Algorithm for computing argmax in Eq. (9)

```
Input: model w, measurements x = (x_1, \ldots, x_T),
        ground-truth classes y^g = (y_1^g, \dots, y_T^g)
Output: predicted classes \bar{y} = (\bar{y}_1, \dots, \bar{y}_T)
Initialize: FP_{max} = FN_{max} = 0
  // NB: []: string concatenation operator,
  // \psi(invalidarg) = NULL, p(NULL) = -\infty
for t = 1 : T  {
 if y_t^g = 0 then \{FP_{max} = FP_{max} + 1\}
  else \{FN_{max} = FN_{max} + 1\}
 for FP = 0 : FP_{max} \{
  for FN = 0 : FN_{max} \{
   if y_t^g = 0 then {
 \psi(FP, FN, y_t = 0) = \operatorname{argmax}(p([\psi(FP, FN, y_{t-1} = 0), 0]),
                                 p([\psi(FP, FN, y_{t-1} = 1), 0]))
\psi(FP, FN, y_t = 1) = \operatorname{argmax}(p([\psi(FP - 1, FN, y_{t-1} = 0), 1]),
                                p([\psi(FP-1, FN, y_{t-1}=1), 1]))
    else {
\psi(FP, FN, y_t = 0) = \operatorname{argmax}(p([\psi(FP, FN - 1, y_{t-1} = 0), 0]),
                                 p([\psi(FP, FN - 1, y_{t-1} = 1), 0]))
\psi(FP, FN, y_t = 1) = \operatorname{argmax}(p([\psi(FP, FN, y_{t-1} = 0), 1]),
                                 // final loop:
best = -\infty, \bar{y} = NULL
for FP = 0 : FP_{max}  {
 for FN = 0 : FN_{max} \{
temp = \max(p(\psi(FP, FN, y_T = 0)),
               p(\psi(FP, FN, y_T = 1))) + \Delta_{F_1}(y^g, FP, FN)
 if temp > best then {
  \bar{y} = \operatorname{argmax}(p(\psi(FP, FN, y_T = 0)),
                  p(\psi(FP, FN, y_T = 1)))
  best = temp \} \} 
return \bar{y}
```

itive samples in the training set. For the SVM module, we used C=0.1 and $\epsilon=0.01$, and a linear kernel. As loss functions for training, we compared a) the zero-one loss, b) the Hamming loss, and c) the F_1 loss based on the algorithm proposed in Section 3. The zero-one loss is the most popular loss function for independent data and the Hamming loss is the most common for structured data. Table 1 reports the results using two different figures of merit: a) accuracy at frame level (equivalent to the Hamming loss measured on the test sequences) and b) the F_1 measure at the sequence level. Given that actions are sparse over time (i.e., there are fewer positive than negative frames in each sequence), the F_1 measure is arguably a more desirable way to describe performance.

Table 1 shows that, in most cases and on average, the best F_1 measure is achieved by training structural SVM with our F_1 loss algorithm, with an average improvement of 8.9 percentage points. This could be expected to some extent given the analogy between the test measure and the training loss. However, Table 1 also shows that, in most cases and on average, training the sequential labeling under the F_1 loss also leads to improvements in the frame-level (Hamming) accuracy over the test set, with an average improvement of 4.3 percentage points. This result shows that the F_1 loss can simply prove more effective for training sequential labeling SVM.

5. CONCLUSION AND FUTURE WORK

In this paper, we have proposed an algorithm for training structural SVM for sequential labeling under an F_1 loss. Such a loss is a more appropriate performance measure than the frame-level accuracy when a balance between precision and recall is sought and is particularly useful in detection scenarios. The proposed algorithm leverages the notion of a state extended with accumulated errors and retains the efficiency of dynamic programming. Experiments conducted on the challenging TUM Kitchen Dataset have reported an improvement of 8.9 percentage points in F_1 measure and 4.3 percentage points in frame-level accuracy over the runner-up (Hamming loss training). In addition, the proposed algorithm can be used for any other loss functions similarly derived from the contingency table of the predicted sequence versus the ground truth. In the near future, we plan on extending the proposed approach to the case of joint multi-label sequential labeling.

6. REFERENCES

- [1] K. Evang, V. Basile, G. Chrupala, and J. Bos, "Elphant: Sequence labeling for word and sentence segmentation," in *Proc. 2013 Conference on Empirical Methods in Natural Language Processing*. IEEE, 2013, pp. 1422–1426.
- [2] N. Nguyen and Y. Guo, "Comparison of sequence labeling algorithms and extensions," in *Proc. ICML*, Corvallis, USA, 2007, p. 681688.
- [3] D. Jurafsky and J. H Martin, *Speech and language processing*, Prentice Hall, New Jersey, USA, 2 edition, 2008.
- [4] E. Zare-Borzeshi, O. Perez-Concha, R. Y. D. Xu, and M. Piccardi, "Joint action segmentation and classification by an extended hidden Markov model," *IEEE Signal Processing Letters*, vol. 20, no. 12, pp. 1207–1210, 2013.
- [5] C. Charron and Y. Hicks, "An evolving MoG for online image sequence segmentation," in *Proc. ICIP*, 2010, pp. 2189–2192.
- [6] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *IEEE Proc.*, vol. 77, pp. 257–286, 1989.
- [7] B. Taskar, C. Guestrin, and D. Koller, "Max-margin Markov networks," in *Proc. NIPS*, 2003.
- [8] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *JMLR*, vol. 6, pp. 1453–1484, 2005.

- [9] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in Proceedings of the 18th European Conference on Machine Learning, 2007, ECML '07, pp. 406–417.
- [10] T. Joachims, "A support vector method for multivariate performance measures," in *Proc. ICML*, Bonn, Germany, 2005, p. 377384.
- [11] M. Tenorth, J. Bandouch, and M. Beetz, "The TUM Kitchen Data Set of Everyday Manipulation Activities for Motion Tracking and Action Recognition," in *Proc. IEEE Int. Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS), in conjunction with ICCV2009*, 2009.
- [12] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. ICML*, 2001, pp. 282–289.
- [13] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proc. ICML*, 2006, pp. 161–168.
- [14] H. Liu, R. Feris, and M. Sun, "Benchmarking human activity recognition," in *Tutorial at CVPR*, Rhode Island, USA, 2012, IEEE.
- [15] T. Joachims, "SVM^{struct}: Support vector machine for complex outputs 3.10," 2008.
- [16] A. Vedaldi, "A MATLAB wrapper of SVM struct," 2011.