Some Progress in Conditional Random Fields

Trevor Cohn,* Andrew Smith† and Miles Osborne†

Universities of Melbourne* and Edinburgh†

June 2005

CRFs: Introduction

- Many speech and language tasks involve predicting a label for an example:
  - Part of speech tagging: examples are sentences, labels are whole tag sequences.
  - Parsing: examples are also sentences, labels are parse trees.
  - Machine translation: examples are source sentences, labels are target sentences.
  - Etc etc.

- Labels can be chains (POS tagging, NER) or trees (parsing); graphs?

CRFs: Introduction

- Typically, labels are decomposed into smaller labels for efficiency reasons:
  - POS tagging: predict a per-word label (rather than an entire sequence).
  - Parsing: predict a local tree (rather than a full parse tree).
  - Etc.
- Stochastic context free grammars / HMMs etc all decompose problems in this manner.
- One consequence: global decision making cannot (easily) be implemented.
- CRFs allow entire label sequences to be predicted in a computationally tractable manner.

CRFs

$$p(y \mid x) = \frac{1}{Z(x)} \exp \sum_{t=1}^{T+1} \sum_{k} \lambda_k f_k(t, y_{t-1}, y_t, x)$$

- λ are the model parameters.
- $\mathbf{f}(\cdot)$ are the feature functions
- T is the length of the sequence.

CRFs: Problems

Two problems with CRFs:

- They do not scale well in terms of the numbers of possible individual labels.
- They overfit the data to a much greater extent than do other approaches.

The rest of this talk with deal with:

- Using error correcting codes to help scale CRFs.
- Using parameter-free methods to regularise CRFs.

Scaling

- Conditional random fields are state-of-the-art models
  - often very slow to train, with high memory requirements
  - particularly expensive for large tasks
  - cost of training dominated by the label set size
- Current engineering solutions
  - supercomputers or large clusters
  - feature induction
  - initial parameters taken from simpler model

Scaling (2)

- We scale CRF estimation to large label sets using error-correcting codes
  - renders multi-class problem as a number of binary problems
  - training a CRF on each binary problem is cheap
- We show that:
  - training time is reduced for current tasks
  - generalisation performance is maintained
  - CRFs can be applied to larger tasks
- Approach compatible with other CRF scaling methods

Outline

- Conditional random fields
- Error-correcting codes
  - classification tasks
  - CRF sequencing tasks
- Experiments and results
- Conclusion and future work

CRF estimation

- Maximise objective (eg. log-likelihood of the labelled training set)
  - numerous methods for optimisation: IIS, GIS, gradient based methods
  - LMVM (L-BFGS) most efficient (Malouf 2002, Wallach 2002)
    - requires repeated calculation of objective and its derivative

$$\mathcal{L} = \sum_i \log p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)})$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_k} = E_{\tilde{p}(\mathbf{x},\mathbf{y})}[f_k] - E_{p(\mathbf{y}|\mathbf{x})}[f_k]$$

CRF estimation (2)

$$E_{p(\mathbf{y}|\mathbf{x})}[f_k] = \sum_i \sum_t \sum_{y'} \sum_y p(Y_{t-1} = y', Y_t = y|\mathbf{x}^{(i)})\, f_k(t, y', y, \mathbf{x}^{(i)})$$

- Computation requires iteration over every possible label pair
  - complexity $O(L^2NTF)$
- We attack the leading L^2 term here, using ECCs

Error-correcting codes

- Allow binary classifiers to be used on multi-class problems
  - ensemble method, k unique classifiers
  - has been applied to text classification (Berger 1999)
- Error-correcting code defines labels used in each binary task

Error-correcting codes (2)

- Training:
  - one classifier trained for each column of coding matrix
- Decoding:
  - each classifier predicts the highest probability class
  - mapped to a label using distance measure: eg. Hamming

Label	Code		
LOC	1	1	0
MISC	0	1	0
ORG	1	0	0
O	0	0	1

ECCs for CRFs

- Identical training step
  - train a binary CRF for each code
  - cheap as there are only two labels
- Decoding performed in three alternative ways:
  - *standalone*
  - *marginals*
  - *product*

Standalone decoding

- Find Viterbi path for each binary CRF; yields sequence of $\{0, 1\}$s
- Create vector of predictions for each time
- Choose closest label vector using Hamming distance

model	time					
1	1	1	1	0	0	0
2	0	1	1	1	0	1
3	0	0	0	1	1	1
...						

Marginals decoding

- Find marginal probabilities at each time for each binary CRF
- Create vector of predictions at each time
- Find closest label vector using L_1 distance

model	time					
1	0.9	0.9	0.6	0.5	0.1	0.4
2	0.2	0.8	0.5	0.2	0.8	0.1
3	0.1	0.1	0.6	0.4	0.8	0.7
...						

Product decoding

- Assume that probability decomposes into the k binary CRFs
- The probability of a labelling is given by:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_P(\mathbf{x})} \prod_j p_j(b_j(\mathbf{y})|\mathbf{x})$$

  - $Z_P(\mathbf{x})$ is a normalising function
  - p_j is the j^{th} binary CRF
  - b_j is the relabelling function for column j of the coding matrix

- This product model is itself a CRF – use standard Viterbi decoding

Experiments: error correcting codes

- Exhaustive
  - contains every unique code
- One-vs-all
  - simplest code, no error-correcting capacity
- Random
  - desirable properties in the limit (Berger, 1999)
- Minimum loss bound
  - better error correcting capacity for common labels

Tasks

- Named entity recognition
  - 8 labels
- Part-of-speech tagging
  - 45 labels
- Combined part-of-speech tagging and noun-phrase chunking
  - 118 labels

Named Entity Recognition

- Tagged entities: location, person, organisation, miscellaneous
- IOB-2 tagging format used
- CoNLL-2003 data set
- 8 labels and 200,000 training tokens

Results: NER performance

- Compared multiclass CRF with coded CRF using *exhaustive* code with 127 binary CRFs
- Same feature sets

Model	Decoding	MLE	Regularised
Multiclass		88.04	89.78
Coded	standalone	88.23*	88.67†
	marginals	88.23*	89.19
	product	**88.69***	**89.69**

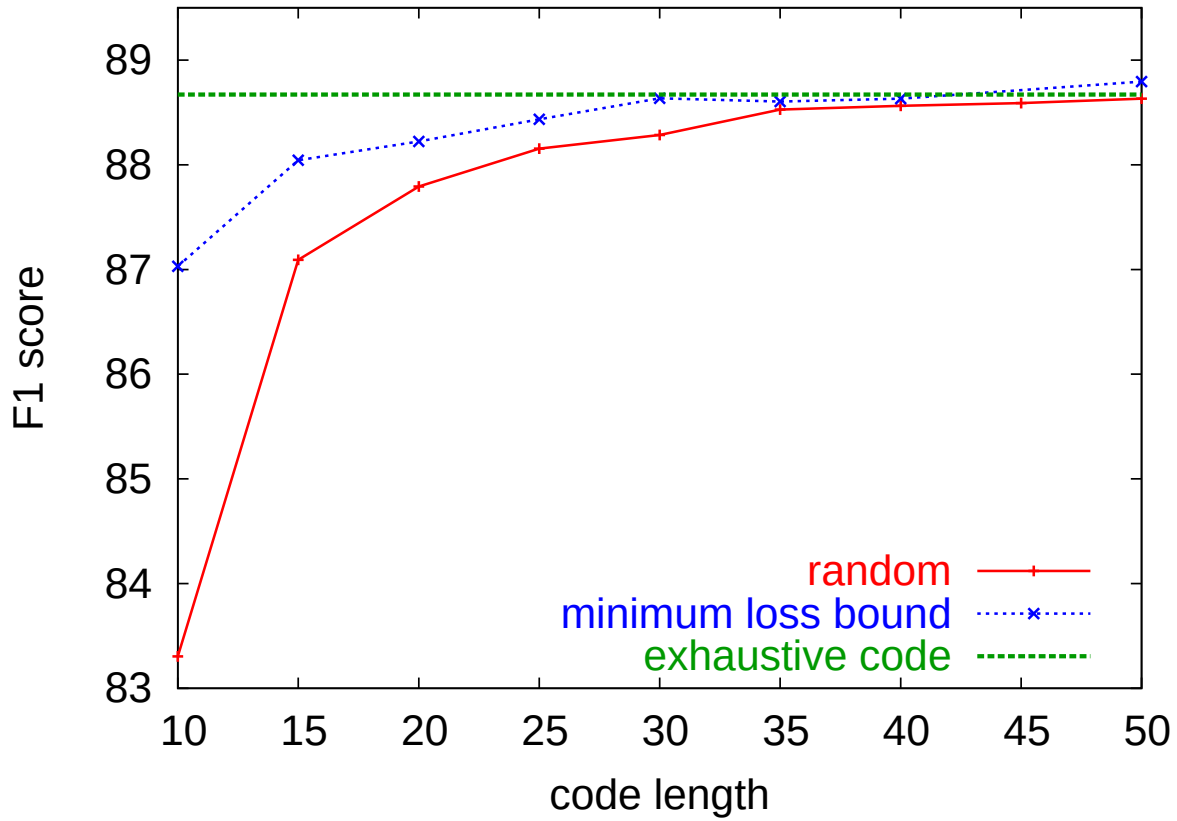
Results: NER timing

- Exhaustive coded CRF training 10 times slower than multiclass
  - exhaustive code is too large (127 columns) for this task
  - partial codes allow much faster training
- One-vs-all code (8 columns) yields equivalent performance, taking less than half the time of the multiclass

NER: alternative codes

- Random codes
  - randomly sample columns from exhaustive code
- Minimum loss bound
  - attempts to find a good subset of columns
  - minimises confusability of commonly occurring labels

NER: alternative codes (2)



POS tagging

- Penn Treebank WSJ
- 45 labels and 1 million training tokens

Results: POS

- Random code of 200 columns and one-vs-all code

Coding	Decoding	MLE	Regularised	Training (hrs)
Coded - 200	standalone	95.63	96.03	293
	marginals	95.68	96.03	
One-vs-all	product	94.90	96.57	25

- Attempt to train multiclass CRF failed
  - were forced to use a small subset of the training data
  - best performance of 95.78
  - full training estimated runtime of more than 1,000 hours

NPC & POS tagging

- Noun phrase chunking – detect whether each token begins or continues a NP, or is outside of a NP
- Simultaneously POS tagging each token
- Derived from CoNLL-2000 data set
- 118 attested labels and 50,000 training tokens

Results: POS & NPC

- Used random code of 200 columns
  - joint tagging accuracy of 90.78% without regularisation
  - training took 100 hours
- Too many labels for tractable multiclass CRF
  - estimated multiclass training of 500 hours or more

Parameter-free smoothing

- Requirement for some form of smoothing when applying CRFs
- Number of techniques to date: priors, feature induction, Bayesian CRFs
- Most popular is Gaussian prior
- Requires fitting hyperparameter(s): “parameterised” method

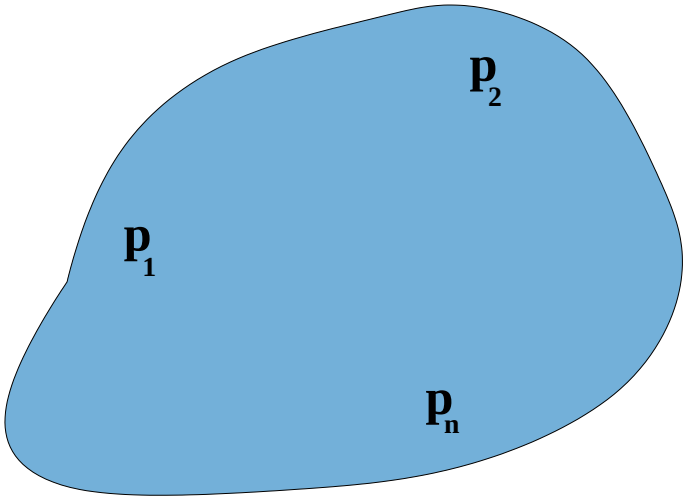
Motivation (2)

- Introduce **logarithmic opinion pools** for CRFs
  - involves model averaging in log-space
  - limits overfitting through variance reduction
  - can be “parameter-free”
- Results competitive with standard regularisation with a prior
- Previous use in other fields: management science (Bordley, 1982)

Outline

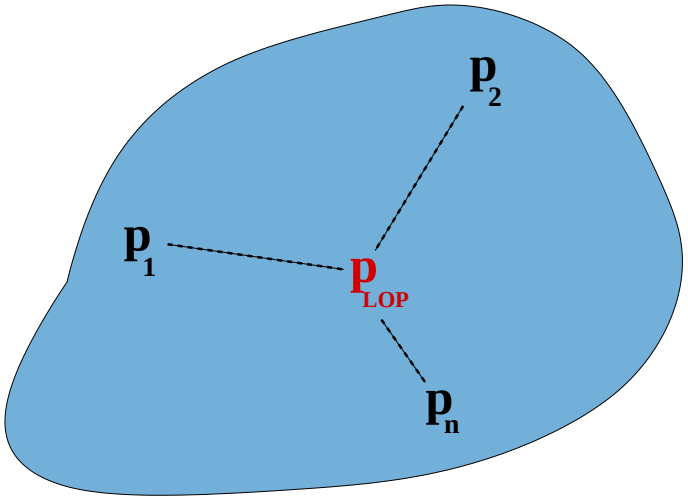
- Logarithmic opinion pools (LOPs)
  - introduction
  - training and decoding
- Experiments and results

Logarithmic Opinion Pools



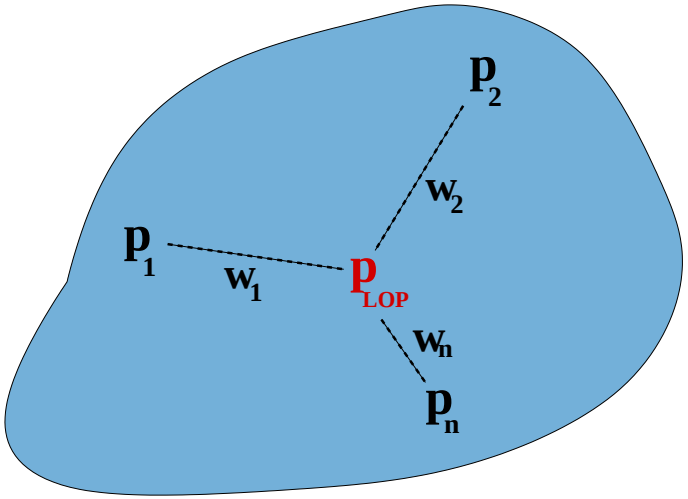
- Pooling of opinions of multiple models
- Each constituent model is an “expert”

Logarithmic Opinion Pools



- Pooling of opinions of multiple models
- Each constituent model is an “expert”

Logarithmic Opinion Pools (2)



- Combines distributions using weighted product
- Weights measure reliability of experts
- Weights specified a priori or found by optimising an objective criterion

Logarithmic Opinion Pools (3)

$$p_{\text{LOP}}(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z_{\text{LOP}}(\mathbf{x})} \prod_{\alpha} p_\alpha(\mathbf{y} \mid \mathbf{x})^{w_\alpha}$$

- $p_\alpha(\mathbf{y} \mid \mathbf{x})$ an expert distribution
- Z_{LOP} a normalisation constant
- Weights w_α non-negative and normalised

Ambiguity Decomposition for LOPs

$$K(q, p_{\text{LOP}}) = \underbrace{\sum_{\alpha} w_\alpha K(q, p_\alpha)}_{\text{E}} - \underbrace{\sum_{\alpha} w_\alpha K(p_{\text{LOP}}, p_\alpha)}_{\text{A}}$$

(Heskes, 1998)

- Trade-off between accuracy (E) and diversity (A)
- Diversity through data set, feature set, training procedure...

LOPs for CRFs

- Multiplicative combination well-suited to CRFs and log-linear models
- LOP is another CRF: LOP-CRF
- Training without approximations
- Decoding using Viterbi, with same complexity as standard CRF
- Avoids problems with additive combination: beam search, etc.

Training LOP-CRFs

- Two-stage training process
- Stage one
  - experts are trained independently, unregularised
- Stage two
  - experts are combined under a LOP
  - LOP weights trained to maximise LOP log-likelihood
  - weights trained unregularised

Tasks: Named Entity Recognition

- CoNLL-2003
- Entities: PER, LOC, ORG, MISC
- Data split
  - training 15,000 sequences
  - development 3,500 sequences
  - test 3,700 sequences

Tasks: Simple POS Tagging

- Derived from CoNLL-2000
- Work developed in parallel to (Cohn et al., 2005)
- 45 tags collapsed to 5 tag categories: N, V, J, R, O (McCallum et al., 2003)
- Split training set
  - training 7,300 sequences
  - development 1,600 sequences
  - test 2,000 sequences

Experiments

- Compare LOP approach to:
  - standard CRF unregularised
  - standard CRF regularised
- Must define:
  - standard CRF with appropriate features
  - set of experts for each LOP: feature subsets

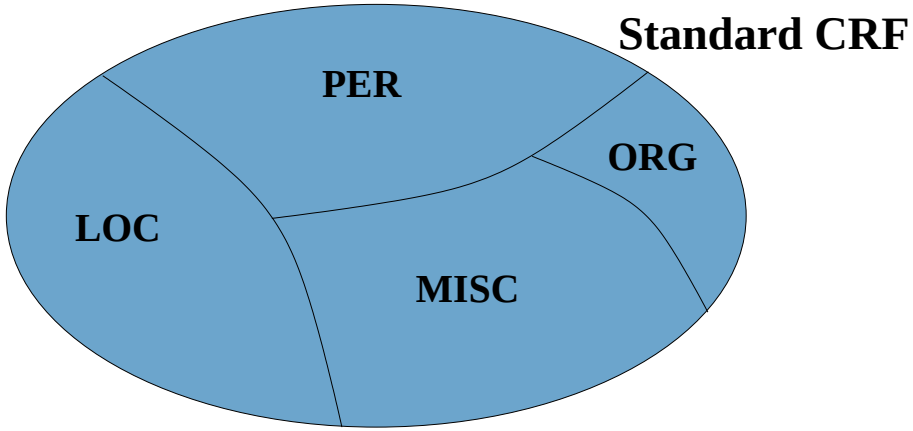
Features: Standard CRF

- NER
  - word and POS tag n-grams
  - orthographic properties: is capitalised, contains a digit, etc.
- POS tagging
  - word n-grams

Features: Expert Sets

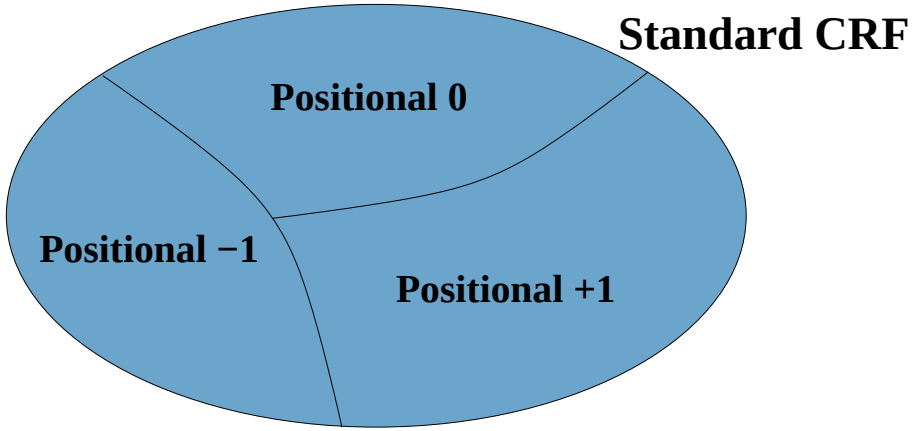
- Partition feature space to create diversity
- Motivate by linguistic intuition
- Expert sets:
  - label
  - positional
  - simple
  - random
- Standard CRF is also an expert

Expert sets: Label



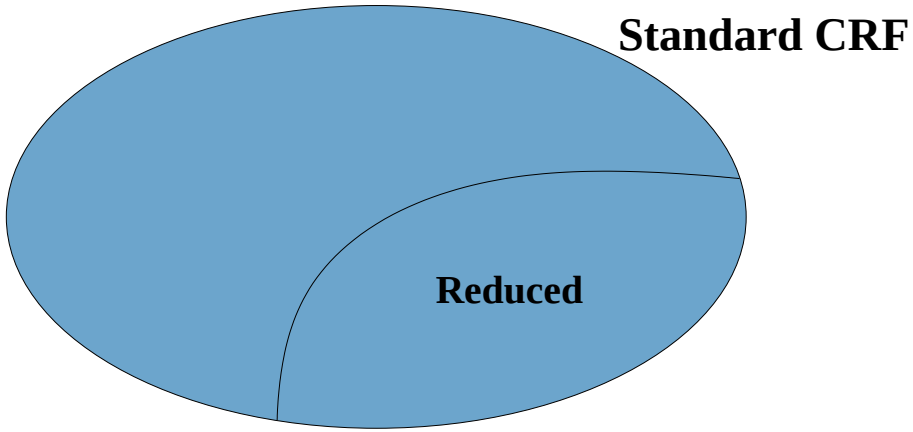
- Features involving specific label on current clique
- Each expert models specific label distribution

Expert sets: Positional



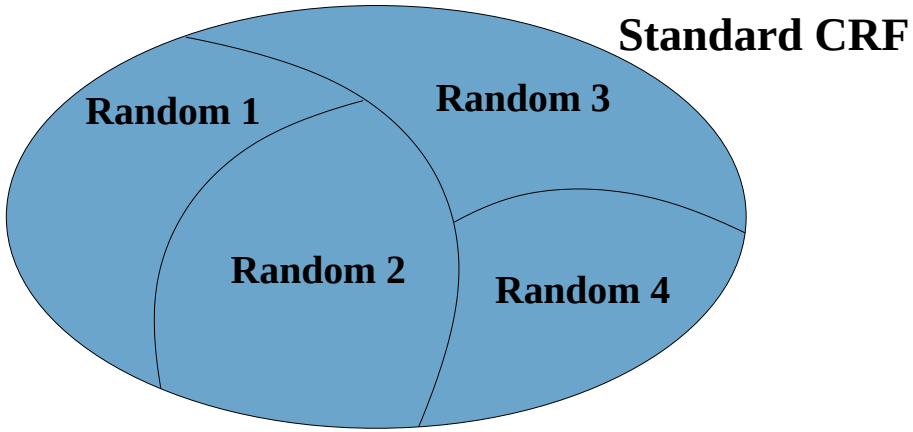
- Features encode events behind, at or ahead of current position
- Each expert models specific positional dependency

Expert sets: Simple



- Reduced subset of standard CRF features
- Models entire distribution

Expert sets: Random



- Random partition of feature space
- Provides a baseline – not linguistically motivated

Baseline Results

NER

Model	F_1
Standard unreg.	88.33
Standard reg.	89.84
Positional -1	73.11
Positional 0	86.96
Positional +1	73.08

POS tagging

Model	Accuracy
Standard unreg.	97.92
Standard reg.	98.02
Random 1	75.23
Random 2	80.88
Random 3	76.99
Random 4	78.36

- 15 runs to obtain regularised score

Trained LOPs: NER

Expert set	Development F_1	Test F_1
Standard unreg.	88.33	81.87
Standard reg.	89.84	83.98
Label	89.30	83.27
Positional	90.35	**84.71***
Simple	90.26	**84.22***
Random	88.84	83.06

- LOPs outperform unregularised standard CRF
- Performance rivals regularised standard CRF
- **Random** achieves good results

Trained LOPs: POS tagging

Expert set	Development accuracy	Test accuracy
Standard unreg.	97.92	97.65
Standard reg.	98.02	97.84
Label	97.99	97.77
Positional	98.03	97.81
Simple	**98.31***	**98.12***
Random	97.99	97.76†

- Similar trends to NER
- Only **random** significantly underperforms regularised standard CRF

Learned Weight Distributions

- **Positional** expert set on NER

Standard	0.45
Positional -1	0.18
Positional 0	0.20
Positional +1	0.17

- Could combine experts under uniform distribution – training required?

Uniform LOPs: POS tagging

Expert set	Uniform Dev. Accuracy	Trained Dev. Accuracy
Label	97.85	97.99
Positional	97.97	98.03
Simple	98.30	98.31†
Random	97.82	97.99

- In most cases outperforms unregularised standard CRF
- Underperforms LOP with trained weights
- Demonstrates learning weights is beneficial

Weight-regularised LOPs

- Possible overfitting without weight regularisation
- Regularising the LOP
  - Very small improvement
  - Number of experts is small: a prior may be beneficial in larger LOPs

Conclusion

- Error-correcting coding can make CRF estimation tractable.
- LOPs enable CRFs to be regularised without optimising hyperparameters.
- Future work:
  - More efficient codes.
  - Co-operative training for LOPs.