Reranking with Linguistic and Semantic Features for Arabic Optical Character Recognition

Nadi Tomeh, Nizar Habash, Ryan Roth, Noura Farra
Center for Computational Learning Systems, Columbia University
{nadi,habash,ryanr,noura}@ccls.columbia.edu

Pradeep Dasigi
Safaba Translation Solutions
pradeep@safaba.com

Mona Diab
The George Washington University
mtdiab@gwu.edu

Abstract

Optical Character Recognition (OCR) systems for Arabic rely on information contained in the scanned images to recognize sequences of characters and on language models to emphasize fluency. In this paper we incorporate linguistically and semantically motivated features to an existing OCR system. To do so we follow an n-best list reranking approach that exploits recent advances in learning to rank techniques. We achieve 10.1% and 11.4% reduction in recognition word error rate (WER) relative to a standard baseline system on typewritten and handwritten Arabic respectively.

1 Introduction

Optical Character Recognition (OCR) is the task of converting scanned images of handwritten, typewritten or printed text into machine-encoded text. Arabic OCR is a challenging problem due to Arabic's connected letter forms, consonantal diacritics and rich morphology (Habash, 2010). Therefore only a few OCR systems have been developed (Märgner and Abed, 2009). The BBN Byblos OCR system (Natajan et al., 2002; Prasad et al., 2008; Saleem et al., 2009), which we use in this paper, relies on a hidden Markov model (HMM) to recover the sequence of characters from the image, and uses an n-gram language model (LM) to emphasize the fluency of the output. For an input image, the OCR decoder generates an n-best list of hypotheses each of which is associated with HMM and LM scores.

In addition to fluency as evaluated by LMs, other information potentially helps in discriminating good from bad hypotheses. For example, Habash and Roth (2011) use a variety of linguistic (morphological and syntactic) and non-linguistic features to automatically identify errors in OCR hypotheses. Another example presented by Devlin et al. (2012) shows that using a statistical machine translation system to assess the difficulty of translating an Arabic OCR hypothesis into English gives valuable feedback on OCR quality. Therefore, combining additional information with the LMs could reduce recognition errors. However, direct integration of such information in the decoder is difficult.

A straightforward alternative which we advocate in this paper is to use the available information to rerank the hypotheses in the n-best lists. The new top ranked hypothesis is considered as the new output of the system. We propose combining LMs with linguistically and semantically motivated features using learning to rank methods. Discriminative reranking allows each hypothesis to be represented as an arbitrary set of features without the need to explicitly model their interactions. Therefore, the system benefits from global and potentially complex features which are not available to the baseline OCR decoder. This approach has successfully been applied in numerous Natural Language Processing (NLP) tasks including syntactic parsing (Collins and Koo, 2005), semantic parsing (Ge and Mooney, 2006), machine translation (Shen et al., 2004), spoken language understanding (Dinarelli et al., 2012), etc. Furthermore, we propose to combine several ranking methods into an ensemble which learns from their predictions to further reduce recognition errors.

We describe our features and reranking approach in §2, and we present our experiments and results in §3.

2 Discriminative Reranking for OCR

Each hypothesis in an n-best list $\{h_i\}_{i=1}^{n}$ is represented by a d-dimensional feature vector $\mathbf{x}_i \in \mathbb{R}^d$. Each $\mathbf{x}_i$ is associated with a loss l_i to generate a labeled n-best list $H = \{(\mathbf{x}_i, l_i)\}_{i=1}^{n}$. The loss is computed as the Word Error Rate (WER) of the

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 549–555,
Sofia, Bulgaria, August 4-9 2013. ©2013 Association for Computational Linguistics

hypotheses compared to a reference transcription. For supervised training we use a set of n-best lists $\mathcal{H} = \{H^{(k)}\}_{k=1}^{M}$.

2.1 Learning to rank approaches

Major approaches to learning to rank can be divided into pointwise score regression, pairwise preference satisfaction, and listwise structured learning. See Liu (2009) for a survey. In this paper, we explore all of the following learning to rank approaches.

Pointwise In the pointwise approach, the ranking problem is formulated as a regression, or ordinal classification, for which any existing method can be applied. Each hypothesis constitutes a learning instance. In this category we use a regression method called Multiple Additive Regression Trees (MART) (Friedman, 2000) as implemented in RankLib.[1] The major problem with pointwise approaches is that the structure of the list of hypotheses is ignored.

Pairwise The pairwise approach takes pairs of hypotheses as instances in learning, and formalizes the ranking problem as a pairwise classification or pairwise regression. We use several methods from this category.

RankSVM (Joachims, 2002) is a method based on Support Vector Machines (SVMs) for which we use only linear kernels to keep complexity low. Exact optimization of the RankSVM objective can be computationally expensive as the number of hypothesis pairs can be very large. Approximate stochastic training strategies reduces complexity and produce comparable performance. Therefore, in addition to RankSVM, we use stochastic sub-gradient descent (*SGDSVM*), Pegasos (*PegasosSVM*) and Passive-Aggressive Perceptron (*PAPSVM*) as implemented in Sculley (2009).[2]

RankBoost (Freund et al., 2003) is a pairwise boosting approach implemented in RankLib. It uses a linear combination of *weak* rankers, each of which is a binary function associated with a single feature. This function is 1 when the feature value exceeds some threshold and 0 otherwise.

RankMIRA is a ranking method presented in (Le Roux et al., 2012).[3] It uses a weighted linear combination of features which assigns the highest score to the hypotheses with the lowest loss. During training, the weights are updated according to the Margin-Infused Relaxed Algorithm (MIRA), whenever the highest scoring hypothesis differs from the hypothesis with the lowest error rate.

In pairwise approaches, the group structure of the n-best list is still ignored. Additionally, the number of training pairs generated from an n-best list depends on its size, which could result in training a model biased toward larger hypothesis lists (Cao et al., 2006).

Listwise The listwise approach takes n-best lists as instances in both learning and prediction. The group structure is considered explicitly and ranking evaluation measures can be directly optimized. The listwise methods we use are implemented in RankLib.

AdaRank (Xu and Li, 2007) is a boosting approach, similar to RankBoost, except that it optimizes an arbitrary ranking metric, for which we use Mean Average Precision (MAP).

Coordinate Ascent (CA) uses a listwise linear model whose weights are learned by a coordinate ascent method to optimize a ranking metric (Metzler and Bruce Croft, 2007). As with AdaRank we use MAP.

ListNet (Cao et al., 2007) uses a neural network model whose parameters are learned by gradient descent method to optimize a listwise loss based on a probabilistic model of permutations.

2.2 Ensemble reranking

In addition to the above mentioned approaches, we couple simple feature selection and reranking models combination via a straightforward ensemble learning method similar to *stacked generalization* (Wolpert, 1992) and *Combiner* (Chan and Stolfo, 1993). Our goal is to generate an overall *meta-ranker* that outperforms all *base-rankers* by learning from their predictions how they correlate with each other.

To obtain the base-rankers, we train each of the ranking models of §2.1 using all the features of §2.3 and also using each feature family added to the baseline features separately. Then, we use the best model for each ranking approach to make predictions on a held-out data set of n-best lists. We can think of each base-ranker as computing one feature for each hypothesis. Hence, the scores generated by all the rankers for a given hypothesis constitute its feature vector.

The held-out n-best lists and the predictions of

[1] http://people.cs.umass.edu/~vdang/ranklib.html

[2] http://code.google.com/p/sofia-ml

[3] https://github.com/jihelhere/adMIRAble

the base-rankers represent the training data for the meta-ranker. We choose RankSVM[4] as the meta-ranker since it performed well as a base-ranker.

2.3 Features

Our features fall into five families.

Base features include the HMM and LM scores produced by the OCR system. These features are used by the baseline system[5] as well as by the various reranking methods.

Simple features ("simple") include the baseline rank of the hypothesis and a 0-to-1 range normalized version of it. We also use a hypothesis confidence feature which corresponds to the average of the confidence of individual words in the hypothesis; "confidence" for a given word is computed as the fraction of hypotheses in the n-best list that contain the word (Habash and Roth, 2011). The more consensus words a hypothesis contains, the higher its assigned confidence. We also use the average word length and the number of content words (normalized by the hypothesis length). We define "content words" as non-punctuation and non-digit words. Additionally, we use a set of binary features indicating if the hypothesis contains a sequence of duplicated characters, a date-like sequence and an occurrence of a specific character class (punctuation, alphabetic and digit).

Word LM features ("LM-word") include the log probabilities of the hypothesis obtained using n-gram LMs with $n \in \{1, \ldots, 5\}$. Separate LMs are trained on the Arabic Gigaword 3 corpus (Graff, 2007), and on the reference transcriptions of the training data (see §3.1). The LM models are built using the SRI Language Modeling Toolkit (Stolcke, 2002).

Linguistic LM features ("LM-MADA") are similar to the word LM features except that they are computed using the part-of-speech and the lemma of the words instead of the actual words.[6]

Semantic coherence feature ("SemCoh") is motivated by the fact that semantic information can be very useful in modeling the fluency of phrases, and can augment the information provided by n-gram LMs. In modeling contextual lexical semantic information, simple bag-of-words models usually have a lot of noise; while more sophisticated models considering positional information have sparsity issues. To strike a balance between these two extremes, we introduce a novel model of semantic coherence that is based on a measure of semantic relatedness between pairs of words. We model semantic relatedness between two words using the Information Content (IC) of the pair in a method similar to the one used by Lin (1997) and Lin (1998).

$$IC(w_1, \mathrm{d}, w_2) = \log \frac{f(w_1, \mathrm{d}, w_2) f(*, \mathrm{d}, *)}{f(w_1, \mathrm{d}, *) f(*, \mathrm{d}, w_2)}$$

Here, d can generally represent some form of relation between w_1 and w_2. Whereas Lin (1997) and Lin (1998) used dependency relation between words, we use distance. Given a sentence, the distance between w_1 and w_2 is one plus the number of words that are seen after w_1 and before w_2 in that sentence. Hence, $f(w_1, \mathrm{d}, w_2)$ is the number of times w_1 occurs before w_2 at a distance d in all the sentences in a corpus. $*$ is a placeholder for any word, i.e., $f(*, d, *)$ is the frequency of all word pairs occurring at distance d. The distances are directional and not absolute values. A similar measure of relatedness was also used by Kolb (2009).

We estimate the frequencies from the Arabic Gigaword. We set the window size to 3 and calculate IC values of all pairs of words occurring at distance within the window size. Since the distances are directional, it has to be noted that given a word, its relations with three words before it and three words after it are modeled. During testing, for each phrase in our test set, we measure semantic relatedness of pairs of words using the IC values estimated from the Arabic Gigaword, and normalize their sum by the number of pairs in the phrase to obtain a measure of Semantic Coherence (SC) of the phrase. That is,

$$SC(p) = \frac{1}{m} \times \sum_{\substack{1 \leq \mathrm{d} \leq W \\ 1 \leq i + \mathrm{d} < n}} IC(w_i, \mathrm{d}, w_{i+\mathrm{d}})$$

where p is the phrase being evaluated, n is the number of words in it, d is the distance between words, W is the window size (set to 3), and m is the number of all possible w_i, $w_{i+\mathrm{d}}$ pairs in the phrase given these conditions.

[4] RankSVM has also been shown to be a good choice for the meta-learner in general stacking ensemble learning (Tang et al., 2010).

[5] The baseline ranking is simply based on the sum of the logs of the HMM and LM scores.

[6] The part-of-speech and the lemmas are obtained using MADA 3.0, a tool for Arabic morphological analysis and disambiguation (Habash and Rambow, 2005; Habash et al., 2009).

	print			hand		
	$\lvert\mathcal{H}_*\rvert$	$\overline{n}$	$\overline{\lvert h\rvert}$	$\lvert\mathcal{H}_*\rvert$	$\overline{n}$	$\overline{\lvert h\rvert}$
$\mathcal{H}_b$	1,560	62	9	2,295	225	8
$\mathcal{H}_m$	1,000	76	9	1,000	225	9
$\mathcal{H}_t$	1,000	64	9	1,000	227	9

Table 1: Data sets statistics. $|\mathcal{H}_*|$ refers to the number of n-best lists, $\overline{n}$ is the average size of the lists, and $\overline{|h|}$ is the average length of a hypothesis.

	print	hand
Baseline	13.8%	35%
Oracle	9.8%	20.9%
Best result	12.4%	30.9%

Table 2: WER for baseline, oracle and best reranked hypotheses.

3 Experiments

3.1 Data and baselines

We used two data sets derived from high-resolution image scans of *typewritten* and *handwritten* Arabic text along with ground truth transcriptions.[7] The BBN Byblos system was then used to process these scanned images into sequences of segments (sentence fragments) and generate a ranked n-best list of hypotheses for each segment (Natajan et al., 2002; Prasad et al., 2008; Saleem et al., 2009). We divided each of the typewritten data set ("print") and handwritten data set ("hand") into three disjoint parts: a training set for the base-rankers $\mathcal{H}_b$, a training set for the meta-ranker $\mathcal{H}_m$ and a test set $\mathcal{H}_t$. Table 1 presents some statistics about these data sets. Our baseline is based on the sum of the logs of the HMM and LM scores. Table 2 presents the WER for our baseline hypothesis, the best hypothesis in the list (our oracle) and our best reranking results which we describe in details in §3.2.

For LM training we used 220M words from Arabic Gigaword 3, and 2.4M words from each "print" and "hand" ground truth annotations.

Effect of n-best training size on WER The size of the training n-best lists is crucial to the learning of the ranking model. In particular, it determines the number of training instances per list. To determine the optimal n to use for the rest of this paper, we conducted the following experiment aims to understand the effect of the size of n-best lists on the reranking performance for one of our best reranking models, namely RankSVM. We trained each model with different sizes of n-best, varying from $n = 5$ to $n = 60$ for "print" data, and between $n = 5$ and $n = 150$ for "hand" data. The top n hypotheses according to the baseline are selected for each n. Figure 1 plots WER as a function of the size of the training list n for both "print" and "hand" data.

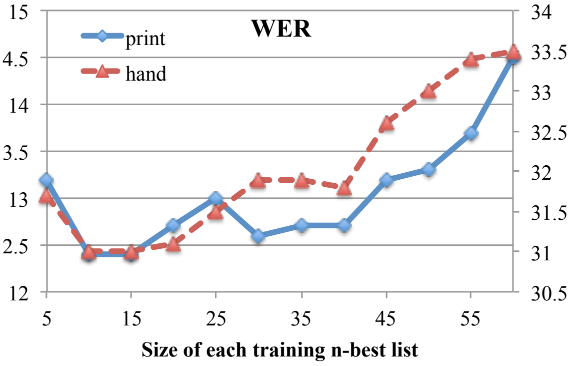

Figure 1: Effect of the size of training n-best lists on WER. The horizontal axis represents the maximum size of the n-best lists and the vertical axis represents WER, left is "print" and right is "hand".

The lowest WER scores are achieved for $n = 10$ and $n = 15$ for both "print" and "hand" data. We note that a small number of hypotheses per list is sufficient for RankSVM to obtain a good performance, but also increasing n further seems to increase the error rate. For the rest of this paper we use the top 10-best hypotheses per segment.

3.2 Reranking results

The reranking results for "print" and "hand" are presented in Table 3. The results are presented as the difference in WER from the baseline WER. See the caption in Table 3 for more information.

For "print", the pairwise approaches clearly outperform the listwise approaches and achieve the lowest WER of 12.4% (10.1% WER reduction relative to the baseline) with 7 different combinations of rankers and feature families. While both approaches do not minimize WER directly, the pairwise methods have the advantage of using objectives that are simpler to optimize, and they are trained on much larger number of examples which may explain their superiority. RankBoost, however, is less competitive with a performance closer to that of listwise approaches. All the methods improved over the baseline with any feature family, except for the pointwise approach which did

[7] The Anfal data set discussed here was collected by the Linguistic Data Consortium.

		Pointwise	Listwise			Pairwise					
	Features	MART	AdaRank	ListNet	CA	RankBoost	RankSVM	SGDSVM	RankMIRA	Pega.SVM	PAPSVM
Print	Base	1.1	-0.4	-1.0	-1.0	-1.0	-1.1	-1.2	-1.2	-1.3	-1.3
	+simple	-0.1	0.0	-0.1	-0.2	0.0	-0.1	0.1	0.0	0.1	0.0
	+LM-word	-1.0	-0.2	0.1	-0.1	-0.1	**-0.3**	**-0.2**	-0.1	0.0	**-0.1**
	+LM-MADA	0.0	-0.3	0.1	-0.2	-0.1	0.0	-0.1	**-0.2**	**-0.1**	**-0.1**
	+SemCoh	0.0	-0.4	0.0	-0.2	-0.1	-0.1	0.0	-0.1	0.0	0.1
	+All	0.6	0.1	0.0	0.1	0.0	0.1	0.2	0.2	0.2	**0.0**
Hand	Base	4.2	-3.1	-3.2	-3.4	-2.9	-3.2	-3.5	-3.8	-3.6	-3.8
	+simple	0.3	-0.1	0.1	0.2	0.1	-0.1	0.2	**-0.2**	0.1	0.2
	+LM-word	0.4	-0.1	0.1	0.8	-0.2	-0.7	-0.2	-0.1	0.0	0.1
	+LM-MADA	0.0	-0.5	0.1	0.0	0.1	-0.4	-0.1	0.3	-0.2	0.1
	+SemCoh	0.0	-0.1	0.0	-0.4	0.0	-0.2	-0.3	**-0.2**	-0.2	0.0
	+All	0.2	0.4	0.0	0.4	0.2	0.4	0.2	0.1	0.2	0.0

Table 3: Reranking results for the “print” and “hand” data sets; the “print” baseline WER is 13.9% and the “hand” baseline WER is 35.0%. The “Base” numbers represent the difference in WER between the corresponding ranker using “Base” features only and the baseline, which uses the same “Base” features. The “+*features*” numbers represent additional gain (relative to “Base”) obtained by adding the corresponding feature family. The “+All” numbers represent the gain of using all features, relative to the best single-family system. The actual WER of a ranker can be obtained by summing the baseline WER and the corresponding “Base” and “+features” scores. Bolded values are the best performers overall.

worse than the baseline. When combined with the “Base” features, “LM-words” lead to improvements with 8 out of 10 rankers, and proved to be the most helpful among feature families. “LM-MADA” follows with improvements with 7 out of 10 rankers. The lowest WER is achieved using one of these two LM-based families. Combining all feature families did not help and in many cases resulted in a higher WER than the best family.

Similar improvements are observed for “hand”. The lowest achieved WER is 31% (11.4% WER reduction relative to the baseline). Here also, the pointwise method increased the WER by 12% relative to the baseline (as opposed to 7% for “print”). Again, the listwise approaches are overall less effective than their pairwise counterparts, except for RankBoost which resulted in the smallest WER reduction among all rankers. The two best rankers correspond to RankMIRA with the “simple” and the “SemCoh” features. The “SemCoh” feature resulted in improvements for 6 out of the 10 rankers, and thus was the best single feature on average for the “hand” data set. As observed with “print” data, combining all the features does not lead to the best performance.

In an additional experiment, we selected the best model for each ranking method and combined them to build an ensemble as described in §2.2. For “hand”, the ensemble slightly outperformed all the individual rankers and achieved the lowest WER of 30.9%. However, for the “print” data, the ensemble failed to improve over the base-rankers and resulted in a WER of 12.4%.

The best overall results are presented in Table 2. Our best results reduce the distance to the oracle top line by 35% for “print” and 29% for “hand”.

4 Conclusion

We presented a set of experiments on incorporating features into an existing OCR system via n-best list reranking. We compared several learning to rank techniques and combined them using an ensemble technique. We obtained 10.1% and 11.4% reduction in WER relative to the baseline for “print” and “hand” data respectively. Our best systems used pairwise reranking which outperformed the other methods, and used the MADA based features for “print” and our novel semantic coherence feature for “hand”.

Acknowledgment

We would like to thank Rohit Prasad and Matin Kamali for providing the data and helpful discussions. This work was funded under DARPA project number HR0011-08-C-0004. Any opinions, findings and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of DARPA. The last two authors, Dasigi and Diab, worked on this project while at Columbia University.

References

Yunbo Cao, Jun Xu, Tie-Yan Liu, Hang Li, Yalou Huang, and Hsiao-Wuen Hon. 2006. Adapting ranking SVM to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '06, pages 186–193.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 129–136.

Philip K. Chan and Salvatore J. Stolfo. 1993. Experiments on multistrategy learning by meta-learning. In *Proceedings of the second international conference on Information and knowledge management*, CIKM '93, pages 314–323.

Michael Collins and Terry Koo. 2005. Discriminative Reranking for Natural Language Parsing. *Comput. Linguist.*, 31(1):25–70, March.

Jacob Devlin, Matin Kamali, Krishna Subramanian, Rohit Prasad, and Prem Natarajan. 2012. Statistical Machine Translation as a Language Model for Handwriting Recognition. In *ICFHR*, pages 291–296.

Marco Dinarelli, Alessandro Moschitti, and Giuseppe Riccardi. 2012. Discriminative Reranking for Spoken Language Understanding. *IEEE Transactions on Audio, Speech & Language Processing*, 20(2):526–539.

Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, December.

Jerome H. Friedman. 2000. Greedy Function Approximation: A Gradient Boosting Machine. *Annals of Statistics*, 29:1189–1232.

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative Reranking for Semantic Parsing. In *ACL*.

David Graff. 2007. Arabic Gigaword 3, LDC Catalog No.: LDC2003T40. Linguistic Data Consortium, University of Pennsylvania.

Nizar Habash and Owen Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan, June.

Nizar Habash and Ryan M. Roth. 2011. Using deep morphology to improve automatic error detection in Arabic handwriting recognition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 875–884.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. MADA+TOKAN: A toolkit for Arabic tokenization, diacritization, morphological disambiguation, POS tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*. The MEDAR Consortium, April.

Nizar Habash. 2010. *Introduction to Arabic Natural Language Processing*. Morgan & Claypool Publishers.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142.

Peter Kolb. 2009. Experiments on the difference between semantic similarity and relatedness. In *Proceedings of the 17th Nordic Conference of Computational Linguistics*, NEALT Proceedings Series Vol. 4.

Joseph Le Roux, Benoit Favre, Alexis Nasr, and Seyed Abolghasem Mirroshandel. 2012. Generative Constituent Parsing and Discriminative Dependency Reranking: Experiments on English and French. In *SP-SEM-MRL 2012*.

Dekang Lin. 1997. Using syntactic dependency as local context to resolve word sense ambiguity. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, EACL '97, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 2*, ACL '98.

Tie-Yan Liu. 2009. *Learning to Rank for Information Retrieval*. Now Publishers Inc., Hanover, MA, USA.

Volker Märgner and Haikal El Abed. 2009. Arabic Word and Text Recognition - Current Developments. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, April. The MEDAR Consortium.

Donald Metzler and W. Bruce Croft. 2007. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, June.

Premkumar Natajan, Zhidong Lu, Richard Schwartz, Issam Bazzi, and John Makhoul. 2002. Hidden Markov models. chapter Multilingual machine printed OCR, pages 43–63. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

Rohit Prasad, Shirin Saleem, Matin Kamali, Ralf Meermeier, and Premkumar Natarajan. 2008. Improvements in hidden Markov model based Arabic OCR. In *Proceedings of International Conference on Pattern Recognition (ICPR)*, pages 1–4.

Shirin Saleem, Huaigu Cao, Krishna Subramanian, Matin Kamali, Rohit Prasad, and Prem Natarajan. 2009. Improvements in BBN's HMM-Based Offline Arabic Handwriting Recognition System. In *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, ICDAR '09, pages 773–777.

D. Sculley. 2009. Large scale learning to rank. In *NIPS 2009 Workshop on Advances in Ranking*.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative Reranking for Machine Translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*, pages 177–184, Boston, Massachusetts, USA, May.

Andreas Stolcke. 2002. SRILM - an Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, volume 2, pages 901–904, Denver, CO.

Buzhou Tang, Qingcai Chen, Xuan Wang, and Xiaolong Wang. 2010. Reranking for stacking ensemble learning. In *Proceedings of the 17th international conference on Neural information processing: theory and algorithms - Volume Part I*, ICONIP'10, pages 575–584.

David H. Wolpert. 1992. Original Contribution: Stacked generalization. *Neural Netw.*, 5(2):241–259, February.

Jun Xu and Hang Li. 2007. AdaRank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 391–398.