# Task Grouping for Multilingual Text Recognition

Jing Huang, Kevin J Liang, Rama Kovvuri, and Tal Hassner

Meta AI
{jinghuang,kevinjliang,ramakovvuri,thassner}@meta.com

**Abstract.** Most existing OCR methods focus on alphanumeric characters due to the popularity of English and numbers, as well as their corresponding datasets. On extending the characters to more languages, recent methods have shown that training different scripts with different recognition heads can greatly improve the end-to-end recognition accuracy compared to combining characters from all languages in the same recognition head. However, we postulate that similarities between some languages could allow sharing of model parameters and benefit from joint training. Determining language groupings, however, is not immediately obvious. To this end, we propose an automatic method for multilingual text recognition with a task grouping and assignment module using Gumbel-Softmax, introducing a task grouping loss and weighted recognition loss to allow for simultaneous training of the models and grouping modules. Experiments on MLT19 lend evidence to our hypothesis that there is a middle ground between combining every task together and separating every task that achieves a better configuration of task grouping/separation.

**Keywords:** OCR, Multilingual Text Recognition, Task Grouping

## 1 Introduction

Optical Character Recognition (OCR) has long been the fundamental task in computer vision. There are many applications such as automatic content extraction for documents [5], translation [43], text style transfer [22] and assistance for robots and visually impaired users. From the perspective of research, OCR can range from relatively easy and controlled tasks such as digit recognition [24], to difficult scenarios such as scene text with arbitrary orientations and shapes [20, 8, 13, 9, 39], and has become an important domain for benchmarking new machine learning techniques.

Nevertheless, most existing OCR approaches focus on numbers and the English alphabet due to English's status as a common *lingua franca* and its subsequent wide availability in popular datasets. Thanks to the introduction of multilingual text detection and recognition datasets and benchmarks [33, 32], there is now a unified platform to measure the model performance on challenging scenarios containing thousands of distinctive characters. Recent methods have shown that training different languages with different recognition heads can improve

end-to-end recognition accuracy compared to combining characters from all languages in the same recognition head [15]. However, it's not clear whether an individual recognition head for each language is optimal. For example, should English and Spanish be separated into two heads? The answer is probably no since they share most of the characters. Moreover, even if separating two languages into two heads does yield the best accuracy, it might not be worth it if the accuracy gain is marginal compared to the increase in number of parameters and/or the inference time. Therefore, one of the questions our work tries to answer is how to decide whether/how languages should be grouped together under the constraint of a limited number of models.

Without any pre-assigned grouping, we treat each of the models at initialization as a generalist agent which looks at all tasks. Then, as the scale tips, each agent is encouraged to be increasingly specialized in one or more tasks; each agent becomes a specialist, each model can still try to learn the other tasks to some lesser extent. Due to different transferability, data variation, and similarities among the tasks, each agent can have different progress in both the specialized tasks and non-specialized tasks, and the specialties will be redistributed automatically as the agents evolve. Eventually, as confirmed by our experiments, this multi-agent system will reach an equilibrium where the specialties for each agent do not change any more, and this is when the task grouping result is finalized.

To summarize, our contributions include:

- To our knowledge, this is *the first work* exploring the grouping of languages for multilingual OCR.
- We propose an automatic grouping mechanism that allows dynamic and differentiable routing of tasks to different heads during training.
- We empirically show that the automatic task grouping model outperforms both the one-task-per-head and the all-tasks-in-one-head baselines. We further show that when the models have different capacities, the task assignment can potentially reflect the underlying task complexity and data distribution.

To promote reproduction of our work, our code is publicly available at https://github.com/facebookresearch/

## 2 Related work

### 2.1 Multilingual text spotting

Text spotting systems combine text detection and text recognition modules to identify the location and content of text in images. Early works approached both modules independently: text proposals for regions containing text were generated first, followed by a recognition module to identify the text given a pre-defined character dictionary. For text detection, state-of-the-art (SotA) methods are mostly based on Region Proposal Networks (RPN) [36], previously proven successful for object detection. Variants of RPNs have been proposed to account for varying text orientations [18], arbitrary shapes [27, 35], and character masking [2]. For text recognition, models typically use an RNN-style design to predict

sequences of characters [41, 14]. Representative methods include connectionist temporal classification (CTC) [12] and attention-based decoders [4, 25, 37].

While earlier systems treated detection and recognition as independent modules, most of the recent works train these modules in an end-to-end manner. Given the inter-dependability of these modules, this training methodology results in performance improvements for these systems. Some representative works include Mask TextSpotter [26], FOTS [28], CharNet [44], etc. For deeper insights into the text spotting systems, we refer the readers to the thorough review in [29]. Similar to these works, we also employ end-to-end training for our system. For recognition, we mainly use an attention-based decoder to make fair comparisons with previous works [27, 15].

With the availability of reliable multilingual datasets such as MLT19 [32], text spotting systems have tried to address the problem of multilingual texts. In addition to detection and recognition modules, some multilingual text spotting systems also include a script identification module [6, 15] to identify the language for text recognition. While text spotting systems such as E2E-MLT [6] and CRAFTS [3] present results for multilingual datasets, they do not explicitly incorporate model specific components adapted for multiple languages. Instead, they combine the characters from all languages to form a larger dictionary for the recognition head. Recently, Multiplexed Multilingual Mask TextSpotter (MMMT) [15] proposed to employ different recognition heads for different scripts, routed through a script identification module at the word level. Unlike MMMT, which employs hard assignment for routing to an appropriate recognition head, we propose to group the languages by training agents to route words to different heads in a data-driven fashion. This automatic grouping mechanism allows for dynamic and differentiable shifting of tasks to optimize the language combinations for various recognition heads.

### 2.2 Multitask learning and grouping

Multitask learning methods have a long history [7, 10]. As their name implies, they jointly learn solutions for multiple tasks, sharing or transferring information between tasks to improve overall performance. Recent deep learning methods assume that the parameters for early layers, which account for low-level feature extraction, are shared among different tasks, while the parameters for later layers, which account for high-level integration of visual signals, are task-specific [47, 16]. Hence, information relevant to all tasks is learned by a shared trunk which later splits to multiple task-specific heads.

A natural question that arises when designing multitask systems is the following: How should tasks be grouped to maximise a model's accuracy? To answer this question, Kang *et al.* [19] proposed learning shared feature representations for related tasks by formulating task grouping as a mixed integer programming problem, where binary indicator variables are used to assign tasks to groups. Unlike this hard group assignment, Kumar *et al.* [23] propose to allow for parameter sharing across groups through soft, latent assignment of task features as a linear combination of a finite number of underlying basis tasks. Zhong *et al.* [48] extend

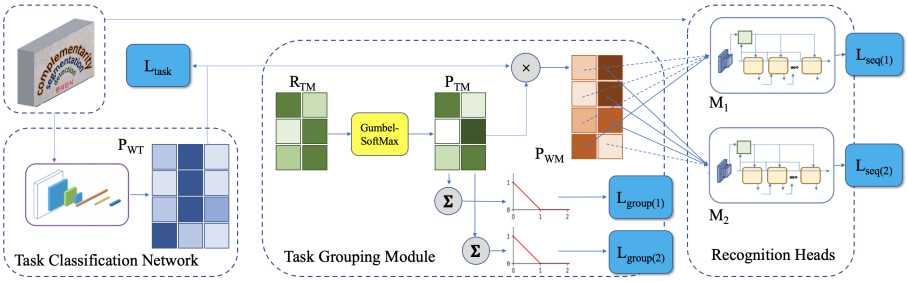


**Fig. 1. Our proposed task grouping framework.** Here we show a batch of 4 inputs potentially belonging to 3 tasks, with 2 recognition heads. See Sec. 3 for more details.

this work by removing constraints on the size of latent basis tasks and adding regularization terms to enforce sparsity in task weights and orthogonality to prohibit commonality among unrelated tasks. Zamir *et al*. [46] proposed a method for modeling the relationship of different visual tasks based on the *transferability* between them. Instead of learning shared representation on a trunk, Strezoski *et al*. [40] introduce a *Task Routing layer* that masks convolutional channels based on the task, effectively creating a sub-network per task.

Our work is similar in spirit to Strezoski *et al*. [40] in that we allow for dynamic routing of tasks to different heads during training. In our approach, however, the routing is done using *Gumbel-Softmax* [17] to ensure probabilistic interpretation of each task and using a novel *grouping loss* for task assignment.

## 3 Methodology

Given a list of tasks $T = \{T_i\}_{1 \leq i \leq t}$ and a list of models $M = \{M_j\}_{1 \leq j \leq m}$, we can define the task grouping of $T$ over $M$ as a mapping $G : T \rightarrow M$, where $G$ is a single-valued function, which means each task will be assigned to exactly one model. On the other hand, $G$ does not need to be an injection, since multiple tasks can be assigned to the same model. $G$ need not be a surjection either, in which case some models will not be assigned with any tasks. Our goal is to find out the best assignment $G$ such that the overall performance is maximized.

Figure 1 shows the core architecture of the proposed task grouping framework. Given an input, which could be a batch of already cropped image patches or pre-extracted features, we first pass them through a *task classification network* that predicts the probabilities of each input belonging to each of the tasks. Under the context of multilingual text recognition, each task $T_i$ can be described as "recognizing the word instance $W_k$ in language set $L_i$", where $W_k$ is the $k$-th instance in a batch of $w$ word crops. We can thus define a probability matrix of size $w \times t$ on the likelihood of each word belonging to each task/language set:

$$P_{WT} = \{p(T_i|W_k)\}_{1 \leq k \leq w, 1 \leq i \leq t} \tag{1}$$

At inference time, $P_{WT}$ can be inferred from a task classification network such as a language prediction network [15]. This is a $t$-way classification problem, and the task classification loss can be computed using a cross entropy loss:

$$L_{task}(W_k) = -\sum_{i=1}^{t} I(T_i = T_{gt}) \log p(T_i|W_k) \tag{2}$$

where $I(T_i = T_{gt})$ is a binary indicator of the task matching the ground truth.

At training time, $P_{WT}$ can be inferred from the ground truth, if there is an annotation of which language each word belongs to: $p(T_i|W_k)$ is 1 if $W_k$ belongs to $T_i$ and 0 otherwise. When the ground truth annotation for the language information is not directly available but the transcription is available, we can make an educated guess of the probability by calculating the proportion of characters in $W_k$ that are supported by language set $L_i$.

### 3.1 Grouping module

Since task-model mapping $G$ is a discrete function, to be able to learn it we can define the following probability matrix, of size $t \times m$:

$$P_{TM} = \{p(M_j|T_i)\}_{1 \leq i \leq t, 1 \leq j \leq m}, \tag{3}$$

where $p(M_j|T_i)$ is the probability of an arbitrary word belonging to $T_i$ to be handled by model $M_j$. Then, we can compute the probability matrix of each word $W_k$ to be handled by model $M_j$ by multiplying $P_{WT}$ and $P_{TM}$:

$$P_{WM} = P_{WT} \cdot P_{TM} \tag{4}$$

Naive task assignment to a group based on traditional SoftMax is a discrete operation and thus non-differentiable. Backpropagation through only the selected task-group pairing would result in high variance gradients leading to unstable learning. Instead, when computing $P_{WM}$ during training, we apply a soft relaxation of the assignment operation using the Gumbel-Softmax [17]. Gumbel-Softmax if fully differentiable with the reparameterization trick and results in gradient backpropagating through all possible task-group pairings, not just the one with the maximum score. We instantiate learnable parameters for task-model assignment as a real-valued matrix $R_{TM} \in \mathbb{R}^{t \times m}$, initialized with all ones (or any equal numbers) in the beginning, and we set the temperature $\tau = 1.0$ throughout the training. At test time, we can just pick the model corresponding to the maximum, *i.e.* the hard mode of Gumbel-Softmax.

### 3.2 Integrated loss

A key difference of our approach compared to [15] is that in our framework, we do not restrict the capability of each model, or recognition head, to support any specific task, *i.e.*, a certain recognition head can only support certain characters,

from the beginning. Instead, we assume each model to be omnipotent in the beginning and has the potential to handle every task. This is necessary since otherwise there is no point in doing the grouping if each model is already designed to do certain tasks.

Therefore, unlike [15], we can directly use the negative log likelihood as the recognition loss $L_{seq(j)}$ for each model $M_j$ without worrying about the unsupported characters:

$$L_{seq} = -\frac{1}{s}\sum_{l=1}^{s}\log p(S_l), \tag{5}$$

where $p(S_l)$ is the predicted probability of character at position $l$ of the sequence, and $s$ is the length of the sequence of character labels. We can, however, perform the pruning at the output layer to remove any characters that do not belong to the task assigned to certain head, once the grouping is determined. This reduces the unnecessary weights in the final model.

The integrated loss across all probabilistic instance-model assignments can thus be calculated as the weighted sum of individual losses:

$$L_{integrated0} = \sum_{k=1}^{w}\sum_{j=1}^{m} p(M_j|W_k)\cdot L_{seq(j)}(W_k, M_j), \tag{6}$$

where the probability term is from $P_{WM}$ of Eq. (4), which is essentially the law of total probability:

$$p(M_j|W_k) = \sum_{i=1}^{t} p(T_i|W_k)\cdot p(M_j|T_i) \tag{7}$$

### 3.3 Integrated loss with a base loss coefficient

With the integrated loss (Eq. (6)), we can see that in general, a task $T_{big}$ with a bigger probability $p(M_j|T_{big})$ to be assigned to a model $M_j$ will contribute a bigger loss than a task $T_{small}$ with a smaller probability $p(M_j|T_{small})$ to be assigned to the model, encouraging the model to optimize towards a better prediction for $T_{big}$, which then encourages $p(M_j|T_{big})$ to be bigger until it reaches 1. A similar but opposite process applies to $p(M_j|T_{small})$, which would become smaller until it reaches 0. As a result, the learned task-model assignment $P_{TM}$ will almost certainly be random and fully depending on the first few iterations due to the positive-feedback loop. We resolve this issue by adding a small positive base loss coefficient, $\epsilon$:

$$L_{integrated} = \sum_{k=1}^{w}\sum_{j=1}^{m}(p(M_j|W_k)+\epsilon)\cdot L_{seq(j)}(W_k, M_j). \tag{8}$$

This ensures that the model not only tries to excel at the tasks assigned to it, but also learns the other tasks at a small but positive rate. The effect of $\epsilon$ can be quantified from the perspective of training data ratios among different tasks.

Assume the original ratio of data from any task is 1, for any model-task pair, the maximum effective data ratio would be $1+\epsilon$, which is achieved when $p$ reaches 1, and the minimum effective data ratio would be $0+\epsilon$, which is achieved when $p$ falls to 0. The ratio $\frac{1+\epsilon}{\epsilon}$ can thus be used to measure how biased the model can potentially be trained towards the most vs. least important task. Based on our ablation study (Sec. 4.4), we set $\epsilon = 0.2$ when training from scratch, $\epsilon = 0.1$ when fine-tuning from pretrained models and $\epsilon = 0$ for the final head-wise fine-tuning.

### 3.4 Grouping loss

While Eq. (8) makes sure that any model has the potential to learn every task, we also would like to ensure that happens within a certain budget, i.e., given the number of different models (heads) we can support, each model is specialized in at least one task. This ensures we do not waste the modeling capacity of an idle head. Therefore, we introduce the following grouping loss

$$L_{group} = \sum_{j=1}^{m} L_{group(j)} = \sum_{j=1}^{m} \max(\mu_j - \sum_{i=1}^{t} p(M_j|T_i), 0), \tag{9}$$

where $\mu_j$ is the least number of tasks model $M_j$ is expected to handle. In most experiments, we set $\mu_j = 1$, meaning that if $M_j$ completely takes over at least one task, the grouping loss for $M_j$ would reach the minimum value 0. Note that the converse does not hold - the grouping loss can reach 0 even when certain model do not excel in any specific task. However, in practice, as long as the number of tasks is larger than or equal to the number of models, the small penalty of the grouping loss could help us achieve the minimum task assignment goal.

## 4 Experimentals

### 4.1 Datasets

Our work leverages a number of public datasets. These sets are summarized in Table 1. We next offer a brief description of these sets.

**ICDAR 2013 dataset (IC13)** [21] This is the oldest set used in this work, originally released for the ICDAR 2013 Robust Reading Competition. It offers 229 training and 233 test images of English text. Text locations are given as axis aligned, rectangular bounding boxes with text annotated at a word level.

**ICDAR 2015 dataset (IC15)** [20] This dataset was introduced in ICDAR'15 and offers more images than IC13: 1000 training and 500 test. Images in this set are of scene text in English, appearing at different orientations, where words are annotated using quadrangle bounding boxes.

**Total Text dataset** [8] This collection offers 1255 training and 300 test, English scene text images. The images reflect a wide range of text orientations and shapes, including curved text examples. To accommodate different shapes, text locations are provided as polygons; recognition labels are given at word level.

**Table 1. Datasets used in our experiments.** #Train: number of training images. Ratio: the relative sampling ratio when the dataset is used in training. Word / Phrase: Annotations given at a word or phrase level. Box type: horizontal, axis aligned (H-Box), arbitrarily rotated (R-Box), quadrangle (Quad), and Polygon. #Lang: Number of languages provided. Note that the Total Text dataset is fully covered in ArT19, and we removed the testing set of Total Text from ArT19.

| Name | #Train | Ratio | Word / Phrase | Box type | #Lang. |
|---|---|---|---|---|---|
| ICDAR13 [21] | 229 | 20 | Word | H-Box | 1 |
| ICDAR15 [20] | 1000 | 20 | Word | Quad | 1 |
| Total Text [8] | 1255 | 50 | Word | Polygon | 1 |
| RCTW17 [38] | 8034 | 20 | Phrase | R-Box | 2 |
| MLT19 [32] | 10000 | 100 | Word | Quad | 10 |
| SynthTextMLT [6] | 252599 | 1 | Word | R-Box | 7 |
| ArT19 [9] | 5303 | 50 | Word | Polygon | 2 |
| LSVT19 [42] | 30000 | 20 | Phrase | Polygon | 2 |

**ICDAR 2017 RCTW dataset (RCTW17)** [38] This set was collected to promote development of OCR methods for in the wild Chinese text. It is partitioned to 8034 and 4229 subsets of training and test images, respectively.

**ICDAR 2019 MLT dataset (MLT19) and SynthTextMLT** [32] was an extension of the ICDAR 2017 MLT dataset (MLT17) [33] for multilingual text detection, recognition and script identification, which contains 10000 training images, 2000 validation images and 10000 test images in 7 different scripts from 10 languages. The dataset contains multi-oriented scene text annotated by quadrangle boxes. A synthetic dataset (SynthTextMLT) [6] containing over 250k synthetic data in 7 scripts was also released along with the MLT19 benchmark. Since MLT19 training and validation sets completely covers the training and validation images in MLT17, though the split is a bit different, we only use MLT19 data for training in this paper.

**ICDAR 2019 ArT dataset (ArT19)** [9] Contains 5603 training and 4563 test images in both English and Chinese, sourced from Total Text [8] and SCUT-CTW1500 [45]. Released as part of the ICDAR 2019 Robust Reading Competition, the images in this collection depict texts in challenging shapes. Similarly to Total Text, text locations are encoded as polygons. We remove all Total Text test images from this set, ensuring that any training on this set can be applied to other sets without risk of test images influencing models trained on this set.

**ICDAR 2019 LSVT dataset (LSVT19)** [42] This is one of the largest data sets used for developing OCR methods: 30000 training and 20000 test images. LSVT images mostly show street views with about 80% of them showing Chinese text and the rest examples in English.

### 4.2 Model training

We base our implementation on the Multiplexer codebase[1]. For fair comparison, we adopt the same segmentation-based detection and ROI mask feature extraction modules as [15], and freeze the pretrained weights of these layers throughout training. For language classification, [15] uses 8 classes including Arabic, Bengali, Chinese, Hindi, Japanese, Korean, Latin, and Symbol, but in our experiment we only use 7 classes by discarding the Symbol class, as it does not have any dedicated dataset and the number of the samples is too small to make a difference.

To expedite the training, we first combine every dataset to train a single recognition head with hidden size 256 and embed size of 200 covering all datasets using the ratios specified in Table 1 for 40k iterations. We then use this weight as a universal pretrained weights for the second stage of training.

Next, we perform a series of experiments that jointly train the grouping module and the recognition heads, each restricting the number of recognition heads to $m$ ($2 \leq m \leq 7$). For each $m$, we launch three training jobs with different random seeds. Each of the training jobs runs for 20k iterations on the MLT19 training datasets only to reduce the potential data imbalance when including the other training set. We record and summarize the final grouping results in Table 2, which we will discuss in 4.3.

Finally, based on the grouping result, we fine-tune each recognition head with only the datasets within the assigned group corresponding to the head. At this stage the grouping is essentially frozen and does not change any more. We can prune the output layer of the decoder so that the characters not belonging to the group are removed, to reduce the parameter number for the final model.

### 4.3 Task grouping results

Table 2 shows the aggregation of grouping results from 18 task grouping experiments with 2 to 7 recognition heads, each repeated for 3 times. All task assignments stabilize after about 10000 iterations.

The top 14 groups are ordered first by the number of occurrences and then by the first occurrence, *i.e.* the minimum number of recognition heads when the group first occurs. All exclusive task-model assignments (one head focusing on one task) occur at least twice, showing the effectiveness of having a dedicated model for each task. Chinese ending up as an individual task occurs in 50% of the cases, which is expected given its high character number and the datasets, except that it's grouped together with Japanese, which shares many characters with it, only once. On the other hand, Hindi seems to be suitable to be grouped with many different languages rather than being trained by itself.

Surprisingly, the most frequent task group that has more than one task is Arabic+Korean, which occurs 5 times. This suggests that there are inherent characteristics shared either by these two scripts, or by the examples in the MLT19 dataset itself, that boost the performance for each other. Another unusual cluster

[1] https://github.com/facebookresearch/MultiplexedOCR

**Table 2. Task grouping result.** Task combinations that end up being grouped together. The 2nd to the 5th columns indicate task names in the final grouping, the number of tasks in the group, the number of occurrences in the 18 experiments, and the minimum number of recognition heads when the combination first occurs.

| Rank | Group | #Tasks within group | #Occurrences | #Heads at first occurrence |
|---|---|---|---|---|
| 1 | Chinese (C) | 1 | 9 | 4 |
| 2 | Latin (L) | 1 | 7 | 5 |
| 3 | Arabic (A) | 1 | 6 | 3 |
| 3 | Korean (K) | 1 | 6 | 3 |
| 5 | Arabic+Korean | 2 | 5 | 2 |
| 6 | Bengali (B) | 1 | 5 | 5 |
| 7 | Japanese (J) | 1 | 4 | 5 |
| 8 | B+C+H+J+L | 5 | 2 | 2 |
| 9 | Hindi+Japanese | 2 | 3 | 3 |
| 10 | Japanese+Latin | 2 | 2 | 4 |
| 11 | Arabic+Hindi | 2 | 2 | 5 |
| 11 | Bengali+Japanese | 2 | 2 | 5 |
| 11 | Hindi+Latin | 2 | 2 | 5 |
| 14 | Hindi (H) | 1 | 2 | 6 |
| 15 | A+K+L | 3 | 1 | 2 |
| 15 | H+J+K | 3 | 1 | 2 |
| 15 | A+B+C+L | 4 | 1 | 2 |
| 15 | B+C+H+J | 4 | 1 | 2 |
| 15 | Chinese+Hindi | 2 | 1 | 3 |
| 15 | Korean+Latin | 2 | 1 | 3 |
| 15 | A+B+J | 3 | 1 | 3 |
| 15 | B+C+L | 3 | 1 | 3 |
| 15 | Arabic+Bengali | 2 | 1 | 4 |
| 15 | Bengali+Hindi | 2 | 1 | 4 |
| 15 | Chinese+Latin | 2 | 1 | 4 |
| 15 | A+B+H | 3 | 1 | 4 |
| 15 | Japanese+Korean | 2 | 1 | 5 |
| 15 | B+C+K | 3 | 1 | 6 |
| 15 | Hindi+Korean | 2 | 1 | 7 |
| 15 | Chinese+Japanese | 2 | 1 | 7 |

is the combination of 5 tasks, Bengali+Chinese+Hindi+Japanese+Latin, which is the only grouping with more than 2 tasks that occurs more than once. We note however that the scattering of the grouping results shows that there can be many local optima for this specific scenario of 7 distinctive scripts. We shall expect higher frequencies of the same grouping results if certain tasks share greater similarity, and we will leave that as one of our future work. We additionally find it interesting that despite a grouping loss to encourage each head to take on at least one task, we observe that some recognition heads might not be assigned with any task in the end when there are 6 or 7 tasks. This means for certain combinations of tasks, training them together could outperform training them separately, even if there is spare resource for a new head.

### 4.4 Ablation study

**Base integrated loss coefficient.** We train the task grouping network with different base integrated loss coefficient $\epsilon$ defined in Eq. (8) on MLT19 training set. The network contains 5 recognition heads that are initialized with the same pretrained weights. We record the number of task assignment changes in

the first 3000 iterations. From Table 3 we can see that, when $\epsilon = 0.0$, there's only 1 assignment change since the model does not have much chance to learn the unassigned tasks; interestingly, when $\epsilon$ is too big (0.3/0.4), there are also fewer changes happening, possibly because there is not much diversity across the models and everything moves in the same direction. The maximum number of assignment changes happen when $\epsilon$ is 0.2 or 0.1. Therefore, in most of our experiments we use 0.2 for early training and 0.1 for fine-tuning.


**Fig. 2. Qualitative results on MLT19 test set [32].** The predicted transcription is rendered with green background, along with the detection confidence, language and the assigned group. The model has 5 heads (groups): group 1 - Arabic (ar) and Hindi (hi), group 2 - Bengali (bn) and Japanese (ja), group 3 - Chinese (zh), group 4 - Latin (la), group 5 - Korean (ko). See Sec. 4.6 for more details.

**Table 3. Ablation study for base integrated loss coefficient $\epsilon$.**

| Base integrated loss coefficient $\epsilon$ | 0.0 | 0.1 | 0.2 | 0.3 | 0.4 |
|---|---|---|---|---|---|
| Assignment changes within 3k iters | 1 | 3 | 5 | 2 | 2 |

**Table 4. Task assignment result for models with different major hyper-parameters.** Each model supports all characters in the beginning so the total number of parameters for each head is high, but they can be pruned when the task assignment stabilizes.

| Embed Size | Hidden Size | Parameter Number | Assigned Task | Charset Size | Final Parameters |
|---|---|---|---|---|---|
| 100 | 224 | 4.05M | Arabic | 80 | 1.15M |
| 150 | 224 | 4.51M | Bengali | 110 | 1.18M |
| 200 | 224 | 4.98M | Japanese | 2300 | 2.13M |
| 100 | 256 | 4.59M | Hindi | 110 | 1.42M |
| 150 | 256 | 5.06M | Korean | 1500 | 2.00M |
| 200 | 256 | 5.52M | Chinese | 5200 | 3.78M |
| 250 | 256 | 5.98M | Latin | 250 | 1.54M |

### 4.5 Task assignment on models with different hyper-parameters

In this section, we perform an interesting experiment that showcases how our design can help assign different tasks to models with different hyper-parameters based on the potential difficulty and the available data. We set the number of models (recognition heads) to be equal to the number of tasks, but set the key hyper-parameters of the models, embed size and hidden size, to be different from each other. We train the overall model on the weighted combination of all datasets listed above, and Table 4 shows the assigned task corresponding to each of the models. We can clearly see the correlation between the number of parameters versus the number of characters in the corresponding character set, with the exception of Latin. This illustrated that in general, when the number of characters grow, the heavier models will outperform lighter models in the long term; however, since Latin words are dominating in all the datasets including many difficult cases like curved text, the task grouping framework learns to spend the heaviest model on it to boost the overall performance.

### 4.6 E2E text recognition

Table 5 shows the results on MLT19 [32] end-to-end multilingual recognition benchmark. Fig. 2 additionally provides qualitative examples of these multilingual. We find that using varying numbers of grouped heads can perform similarly to (and in some cases, better than) the multiplexed approach of a separate recognition head per language [15]. This is an interesting result, as it means we can significantly cut down on the computational cost and model size with little impact or even some gains to the performance. Notably, we also find that increasing the number of heads from a single shared head (Mask TextSpotter V3 [27]) to even just two grouped heads leads to a significant increase in F1-score.

We provide qualitative failure cases in Fig. 3. While detection errors could be attributed to arbitrary text shape, blurred text, glossy surfaces and rare

**Table 5. End-to-end recognition results on MLT19.** Note that there are two versions results of CRAFTS, one from the official MLT19 website and one from paper [3]. Importantly, CRAFTS has a ResNet-based feature extraction which is much bigger than the one with 5-Convs used in our experiments.

| Method | F | P | R |
|---|---|---|---|
| E2E-MLT [6] | 26.5 | 37.4 | 20.5 |
| RRPN+CLTDR [30] | 33.8 | 38.6 | 30.1 |
| CRAFTS [3] | 51.7 | 65.7 | 42.7 |
| CRAFTS (paper) [3] | **58.2** | **72.9** | **48.5** |
| Mask TextSpotter V3 (1 head) [27] | 39.7 | **71.8** | 27.4 |
| Multiplexed TextSpotter (8 heads) [15] | 48.2 | 68.0 | 37.3 |
| Grouped (2 heads) | 45.5 | 67.7 | 34.3 |
| Grouped (3 heads) | 47.1 | 67.0 | 36.3 |
| Grouped (4 heads) | 47.9 | 66.7 | 37.4 |
| Grouped (5 heads) | **48.5** | 67.7 | **37.8** |
| Grouped (6 heads) | 48.3 | 67.8 | 37.5 |
| Grouped (7 heads) | 48.2 | 68.0 | 37.3 |

fonts; recognition errors could be attributed to text ordering, text resolution and challenging scripts.

## 5 Conclusions

Text is one of the most ubiquitous visual object classes in real-world scenes, making understanding it practical and critically important. Processing multiple languages, however, requires substantial resources, to accurately recognize the subtleties of appearances variations of different scripts and different intra-script characters. This ability was, therefore, previously accomplished by specializing separate network heads to specific languages. We, instead, are the first to propose automatically grouping different languages together, in the same recognition heads. Our dynamic, and differentiable task shifting approach automatically routes tasks to different heads while the network trains, optimizing for the best, bottom line accuracy across all languages. Extensive tests show our method to not only achieve SotA accuracy, but to do so with fewer recognition heads and hyperparameters, consequently making is a practical design choice for real-world OCR systems.

**Future work** Our work leaves several natural follow-up directions. One interesting question relates to the scalability of our approach: How many multitask heads, for example, would be required to effectively learn hundreds of languages? Another intriguing direction is extending our multitask learning and task grouping to include neural architecture search as part of its design. Such a solution should allow growing heads with different architectures for different languages to account for, *e.g.*, harder vs. easier languages. Finally, another potential extension could be continual language learning [34]: adding more languages as relevant training data becomes available, without retraining or regrouping existing languages. Alternative grouping approaches based on Bayesian nonparametric approaches like the Chinese Restaurant Process [1] or Indian Buffet Process [11, 31] may be natural ways to perform groupings in such settings.


**Fig. 3. Error analysis on MLT19.** Detection errors are represented in red outline and recognition errors in purple. See Sec. 4.6.

## References


1. Aldous, D.J.: Exchangeability and related topics. In: École d'Été de Probabilités de Saint-Flour XIII — 1983 (1985)
2. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character region awareness for text detection. In: IEEE Conf. Comput. Vis. Pattern Recog. (2019)
3. Baek, Y., Shin, S., Baek, J., Park, S., Lee, J., Nam, D., Lee, H.: Character region attention for text spotting. In: Eur. Conf. Comput. Vis. (2020)
4. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
5. Burie, J.C., Chazalon, J., Coustaty, M., Eskenazi, S., Luqman, M.M., Mehri, M., Nayef, N., Ogier, J.M., Prum, S., Rusiñol, M.: Icdar2015 competition on smartphone document capture and ocr (smartdoc). In: International Conference on Document Analysis and Recognition (ICDAR) (2015)
6. Bušta, M., Patel, Y., Matas, J.: E2e-mlt-an unconstrained end-to-end method for multi-language scene text. In: ACCV (2018)
7. Caruana, R.: Multitask learning. Machine learning (1997)
8. Chng, C.K., Chan, C.S.: Total-text: A comprehensive dataset for scene text detection and recognition. In: Int. Conf. on Document Anal. and Recog. (2017)
9. Chng, C.K., Liu, Y., Sun, Y., Ng, C.C., Luo, C., Ni, Z., Fang, C., Zhang, S., Han, J., Ding, E., Liu, J., Karatzas, D., Chan, C.S., Lianwen, J.: Icdar2019 robust reading challenge on arbitrary-shaped text-rrc-art. In: Int. Conf. on Document Anal. and Recog. (2019)
10. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: Proc. int. conf. on Knowledge discovery and data mining (2004)
11. Ghahramani, Z., Griffiths, T.L.: Infinite Latent Feature Models and the Indian Buffet Process. Neural Information Processing Systems (2006)
12. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Int. Conf. Mach. Learning. (2006)
13. Hassner, T., Rehbein, M., Stokes, P., Wolf, L.: Computation and palaeography: potentials and limits (dagstuhl perspectives workshop 12382). Dagstuhl Manifestos (2013)
14. He, P., Huang, W., Qiao, Y., Loy, C.C., Tang, X.: Reading scene text in deep convolutional sequences. In: AAAI (2016)
15. Huang, J., Pang, G., Kovvuri, R., Toh, M., Liang, K.J., Krishnan, P., Yin, X., Hassner, T.: A multiplexed network for end-to-end, multilingual ocr. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021)
16. Inkawhich, N., Liang, K., Carin, L., Chen, Y.: Transferable perturbations of deep feature distributions. In: Int. Conf. Learn. Represent. (2020)
17. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
18. Jiang, Y., Zhu, X., Wang, X., Yang, S., Li, W., Wang, H., Fu, P., Luo, Z.: R2cnn: rotational region cnn for orientation robust scene text detection. arXiv preprint arXiv:1706.09579 (2017)
19. Kang, Z., Grauman, K., Sha, F.: Learning with whom to share in multi-task feature learning. In: Int. Conf. Mach. Learning. (2011)
20. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., et al.: Icdar 2015 competition on robust reading. In: Int. Conf. on Document Anal. and Recog. (2015)

21. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazan, J.A., De Las Heras, L.P.: Icdar 2013 robust reading competition. In: Int. Conf. on Document Anal. and Recog. (2013)
22. Krishnan, P., Kovvuri, R., Pang, G., Vassilev, B., Hassner, T.: Textstylebrush: transfer of text aesthetics from a single example. arXiv preprint arXiv:2106.08385 (2021)
23. Kumar, A., Daumé III, H.: Learning task grouping and overlap in multi-task learning. In: Int. Conf. Mach. Learning. (2012)
24. LeCun, Y.: The mnist database of handwritten digits. http://yann. lecun. com/exdb/mnist/ (1998)
25. Lee, C.Y., Osindero, S.: Recursive recurrent nets with attention modeling for ocr in the wild. In: IEEE Conf. Comput. Vis. Pattern Recog. (2016)
26. Liao, M., Lyu, P., He, M., Yao, C., Wu, W., Bai, X.: Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. IEEE Trans. Pattern Anal. Mach. Intell. (2021)
27. Liao, M., Pang, G., Huang, J., Hassner, T., Bai, X.: Mask textspotter v3: Segmentation proposal network for robust scene text spotting. In: European Conference on Computer Vision (2020)
28. Liu, X., Liang, D., Yan, S., Chen, D., Qiao, Y., Yan, J.: Fots: Fast oriented text spotting with a unified network. In: IEEE Conf. Comput. Vis. Pattern Recog. (2018)
29. Long, S., He, X., Yao, C.: Scene text detection and recognition: The deep learning era. Int. J. Comput. Vis. (2020)
30. Ma, J., Shao, W., Ye, H., Wang, L., Wang, H., Zheng, Y., Xue, X.: Arbitrary-oriented scene text detection via rotation proposals. IEEE Trans. Multimedia (2018)
31. Mehta, N., Liang, K., Verma, V.K., Carin, L.: Continual learning using a bayesian nonparametric dictionary of weight factors. In: International Conference on Artificial Intelligence and Statistics (2021)
32. Nayef, N., Patel, Y., Busta, M., Chowdhury, P.N., Karatzas, D., Khlif, W., Matas, J., Pal, U., Burie, J.C., Liu, C.l., Ogier, J.M.: Icdar2019 robust reading challenge on multi-lingual scene text detection and recognition—rrc-mlt-2019. In: Int. Conf. on Document Anal. and Recog. (2019)
33. Nayef, N., Yin, F., Bizid, I., Choi, H., Feng, Y., Karatzas, D., Luo, Z., Pal, U., Rigaud, C., Chazalon, J., et al.: Icdar2017 robust reading challenge on multi-lingual scene text detection and script identification-rrc-mlt. In: Int. Conf. on Document Anal. and Recog. (2017)
34. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual Lifelong Learning with Neural Networks: A Review. Neural Networks (2019)
35. Qin, S., Bissacco, A., Raptis, M., Fujii, Y., Xiao, Y.: Towards unconstrained end-to-end text spotting. In: Int. Conf. Comput. Vis. (2019)
36. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Adv. Neural Inform. Process. Syst. (2015)
37. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: IEEE Conf. Comput. Vis. Pattern Recog. (2016)
38. Shi, B., Yao, C., Liao, M., Yang, M., Xu, P., Cui, L., Belongie, S., Lu, S., Bai, X.: Icdar2017 competition on reading chinese text in the wild (rctw-17). In: Int. Conf. on Document Anal. and Recog. (2017)
39. Singh, A., Pang, G., Toh, M., Huang, J., Galuba, W., Hassner, T.: TextOCR: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021)

40. Strezoski, G., van Noord, N., Marcel, W.: Learning task relatedness in multi-task learning for images in context. In: Proc. Int. Conf. on Multimedia Retrieval (2019)
41. Su, B., Lu, S.: Accurate scene text recognition based on recurrent neural network. In: ACCV (2014)
42. Sun, Y., Ni, Z., Chng, C.K., Liu, Y., Luo, C., Ng, C.C., Han, J., Ding, E., Liu, J., Karatzas, D., Chan, C.S., Jin, L.: Icdar 2019 competition on large-scale street view text with partial labeling – rrc-lsvt. In: Int. Conf. on Document Anal. and Recog. (2019)
43. Toyama, T., Sonntag, D., Dengel, A., Matsuda, T., Iwamura, M., Kise, K.: A mixed reality head-mounted text translation system using eye gaze input. In: Proceedings of the 19th international conference on Intelligent User Interfaces (2014)
44. Xing, L., Tian, Z., Huang, W., Scott, M.R.: Convolutional character networks. In: Int. Conf. Comput. Vis. (2019)
45. Yuliang, L., Lianwen, J., Shuaitao, Z., Sheng, Z.: Detecting curve text in the wild: New dataset and new solution. arXiv preprint arXiv:1712.02170 (2017)
46. Zamir, A.R., Sax, A., Shen, W., Guibas, L.J., Malik, J., Savarese, S.: Taskonomy: Disentangling task transfer learning. In: IEEE Conf. Comput. Vis. Pattern Recog. (2018)
47. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Eur. Conf. Comput. Vis. (2014)
48. Zhong, S., Pu, J., Jiang, Y.G., Feng, R., Xue, X.: Flexible multi-task learning with latent task grouping. Neurocomputing (2016)