UNSUPERVISED DOMAIN ADAPTATION SCHEMES FOR BUILDING ASR IN LOW-RESOURCE LANGUAGES

Anoop C S[1], *Prathosh A P*[2], *A G Ramakrishnan*[1]

[1]Indian Institute of Science, Bengaluru, India
[2]Indian Institute of Technology, Delhi, India

ABSTRACT

Building an automatic speech recognition (ASR) system from scratch requires a large amount of annotated speech data, which is difficult to collect in many languages. However, there are cases where the low-resource language shares a common acoustic space with a high-resource language having enough annotated data to build an ASR. In such cases, we show that the domain-independent acoustic models learned from the high-resource language through unsupervised domain adaptation (UDA) schemes can enhance the performance of the ASR in the low-resource language. We use the specific example of Hindi in the source domain and Sanskrit in the target domain. We explore two architectures: i) domain adversarial training using gradient reversal layer (GRL) and ii) domain separation networks (DSN). The GRL and DSN architectures give absolute improvements of 6.71% and 7.32%, respectively, in word error rate over the baseline deep neural network model when trained on just 5.5 hours of data in the target domain. We also show that choosing a proper language (Telugu) in the source domain can bring further improvement. The results suggest that UDA schemes can be helpful in the development of ASR systems for low-resource languages, mitigating the hassle of collecting large amounts of annotated speech data.

Index Terms— unsupervised domain adaptation, speech recognition, low-resource language, ASR, domain separation networks.

1. INTRODUCTION

Advancements in deep learning have brought performance improvements in acoustic and language modeling, yielding robust automatic speech recognition (ASR) systems in many languages. However, such systems require a large amount of speech data and the associated transcriptions. It is tough to collect large volumes of paired speech and transcriptions for most low-resource languages. It is estimated that only about 1% of the world languages have the minimum amount of data that is needed to train an ASR [1]. However, in many cases, especially for languages in south Asia, we can find a "close enough" language with the same set (or a superset) of phonemes as the low-resource language and enough resources for building an ASR. In this work, we show that a better performing ASR can be built for the low-resource language using unsupervised domain adaptation (UDA) of acoustic models from the corresponding high-resource language. This method has the benefit of modeling on real data in comparison to the data augmentation techniques like vocal tract length perturbation [2, 3], speech and tempo perturbation [4], noise addition [5], data synthesis [6], and spectral augmentation [7], where the modeling makes use of the synthetic data as well.

Unsupervised domain adaptation (UDA) has been successfully applied to various tasks to alleviate the shift between the train and test distributions. [8] shows good adaptation performance in the classification task on digit image datasets having considerable domain shifts. They learn features that are discriminative for the image classification task and invariant to the domain. They introduce a gradient reversal layer (GRL) for achieving this objective. [9] reports performance improvements in speech recognition for data shifted in the domain by gender and accent. [10] employs GRL layers to reduce the mismatch between train and test domains in the task of emotion recognition from speech data. [11] uses the GRL approach to improve the word error rate (WER) in speech recognition with the source domain data as clean speech and the target domain data as contaminated speech. They also show the robustness of the approach to the domain shifts caused by the differences in datasets.

The basic UDA scheme with a GRL tries to learn domain invariant features but ignores the individual characteristics of each domain. [12] introduces domain separation networks (DSN) and shows improvements in a range of UDA scenarios in the image classification task. They learn two representations: one specific to each domain and the other common to both domains. [13] uses DSN for adaptation from clean speech to noisy speech.

In this work, we explore the feasibility of the UDA schemes in the ASR task on low-resource languages that share the same acoustic space with a high-resource language. Specifically, we place the following assumptions in the selection of high and low resource language pairs.

1. The acoustic space spanned by the low-resource language is a subspace of that of the high-resource language. This requires the phoneme set of the low-resource language to be a subset of the high-resource language.

2. There exists a high-resource language with reasonable amount of paired audio data and transcriptions to build an ASR.

3. The low-resource language has enough text data available to train the language models.

4. The speech data available in the low-resource language is quite limited, and the transcriptions are not available.

Though the above assumptions seem quite constrictive, we can easily find a few low-resource languages in the Indian subcontinent which share a common acoustic space with a reasonably high-resource language. In this work, we use Hindi as the high-resource language and Sanskrit as the low-resource language, both belonging to the Indo-Aryan language family. Hindi is written in the Devanagari script, and many of its words are derived from Sanskrit. However, there exist substantial differences in the vocabulary and pronunciation between the two languages. One of the important differences is that the schwa ('ə'), implicit in each consonant of the script, is not pronounced at the end of words and in some other contexts in Hindi. The script does not tell us when the schwa should be deleted, a phenomenon known as schwa deletion. For example, the word नमकीन (meaning salty) is pronounced as *nam'kīn* in Hindi and not *namakīna*. Another difference is the pitch accents that are common in Sanskrit. Despite the above differences, these languages share a common phoneme set. So we can intuitively argue that there exists a domain shift between the distributions of acoustic features in Hindi and Sanskrit. This makes us believe that the speech recognition problem in Sanskrit, an extremely low-resource language, may be posed as an UDA problem from Hindi, a language with a reasonably good collection of annotated audio.

2. UNSUPERVISED DOMAIN ADAPTATION FOR ACOUSTIC MODELING

We pose the problem of acoustic modeling in Sanskrit as an unsupervised domain adaptation task from Hindi. We build a deep neural network (DNN) - hidden Markov model (HMM) ASR system [14] for Sanskrit with the domain-independent acoustic models learned from Hindi through UDA approaches. We make use of the UDA schemes introduced in [8] and [12]. [15–18] have also addressed the ASR problem in Sanskrit, but in all cases, acoustic models are built solely using the speech data from Sanskrit.

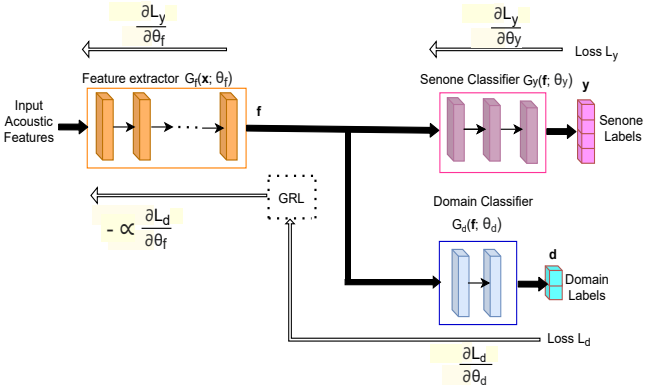

Fig. 1: Block diagram of the basic scheme for unsupervised domain adaptation with a gradient reversal layer (GRL). θ_f, θ_y and θ_d represent the parameters of the feature extractor, senone classifier, and domain classifier, respectively.

2.1. Adversarial training using GRL

In GRL-based adversarial training, we try to learn a feature representation invariant to the domain, but good enough in discriminating the senone labels. The DNN architecture for learning the acoustic model consists of three parts: feature extractor, senone classifier, and domain classifier. A block diagram of the UDA architecture employing GRL is shown in Fig. 1. Feature extractor G_f maps the input acoustic features $\mathbf{x}$ to an internal representation $\mathbf{f} \in \mathcal{R}^D$. Senone classifier G_y maps the output of the feature extractor to the senone labels $\mathbf{y}$ whereas the domain classifier G_d maps them to domain labels $\mathbf{d}$.

We train the network to minimize the senone classification loss during the training phase by optimizing the parameters in the feature extractor G_f and senone classifier G_y. This makes the network look for features that are capable of discriminating the senone labels. To make the features domain-invariant, we optimize the parameters of the feature extractor to maximize the domain classification loss. But at the same time, the parameters of the domain classifier G_d are optimized to minimize the domain classification loss. This is achieved by introducing a GRL between the feature extractor G_f and domain classifier G_d. During the forward pass, GRL acts as an identity transform. GRL takes the gradient from the subsequent layer during the backward pass, multiplies it

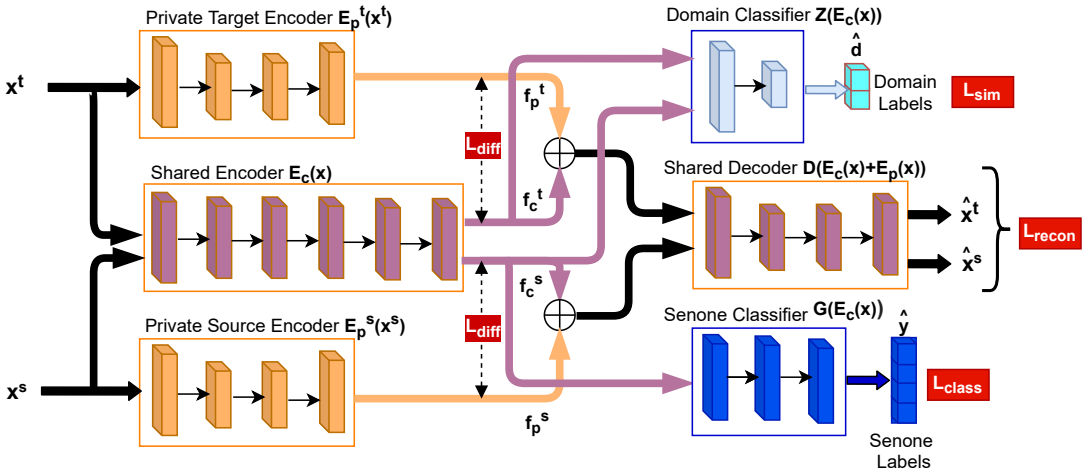

Fig. 2: Block diagram of the domain separation network to model private and shared components.

with $-\alpha$, a hyperparameter to control the trade-off between senone discrimination and domain-invariance, and passes it to the preceding layer.

During the inference time, the domain classifier and GRL are ignored. The acoustic feature vectors are passed through the feature extractor and the senone classifier, and senone labels are predicted.

2.2. Domain separation networks

A block diagram of the DSN architecture is shown in Fig. 2. They model both the private and shared components of the domain representation. Source domain is where transcribed audio data is available to train acoustic models in a domain-invariant manner. These models are used in the target domain for inference. Private encoders $E_p^t(.)$ and $E_p^s(.)$ extract components f_p^t and f_p^s which are specific to the target and source domains. Shared encoder $E_c(.)$ is common to both domains and extract the shared components f_c^t and f_c^s . Shared decoder D tries to reconstruct the input using the private and shared components. $G(.)$, the senone classifier maps f_c^s to the senone label $\widehat{\mathbf{y}}$. The domain classifier $Z(.)$ maps the shared components f_c^s and f_c^t to their respective domain labels $\widehat{\mathbf{d}}$.

The network is trained to minimize the following loss function with respect to the parameters of E_p, E_c, G and Z:

$$L = L_{class} + \beta L_{sim} + \gamma L_{diff} + \delta L_{recon} \qquad (1)$$

where β, γ, and δ are hyperparameters. L_{class} represents the senone classification loss and is applied only to the source domain. It is computed as the negative log-likelihood of the ground-truth senone labels. L_{sim} represents the domain adversarial similarity loss and is computed as the negative log-likelihood of the domain labels. L_{sim} ensures that the shared components f_c^t and f_c^s are as similar as possible irrespective of their domain so that the domain classifier cannot reliably predict the domain of the sample from its shared representation. Parameters of the domain classifier $Z(.)$ are trained to minimize the domain classification loss while the parameters of the shared encoder E_c are trained to maximise the domain classification loss. This is also accomplished with a GRL. L_{diff} encourages the shared component f_c and the private component f_p to encode different aspects of the input. This is achieved by imposing a soft subspace orthogonality constraint between the private and shared components.

$$L_{diff} = \|\mathbf{F}_c^{sT}\mathbf{F}_p^s\|_F^2 + \|\mathbf{F}_c^{tT}\mathbf{F}_p^t\|_F^2 \qquad (2)$$

where $\mathbf{F}_c^s$, $\mathbf{F}_c^t$, $\mathbf{F}_p^s$ and $\mathbf{F}_p^t$ are matrices with f_c^s, f_c^t, f_p^s and f_p^t as rows. $\|.\|_F$ denotes the Frobenius norm. L_{recon} is the reconstruction loss, computed as the mean squared error (MSE) between $\mathbf{x}$ and $\hat{\mathbf{x}} = D(E_c(\mathbf{x}) + E_p(\mathbf{x}))$.

$$L_{recon} = \sum_{i=1}^{N_s} \|\mathbf{x}_i^s - \hat{\mathbf{x}}_i^s\|^2 + \sum_{i=1}^{N_t} \|\mathbf{x}_i^t - \hat{\mathbf{x}}_i^t\|^2 \qquad (3)$$

where N_s and N_t represent the number of speech frames from the source and target domains. We also validate the performance of the system with scale-invariant mean squared error (SIMSE) which is computed as:

$$L_{SIMSE} = \frac{1}{k}\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 - \frac{1}{k^2}([\mathbf{x} - \hat{\mathbf{x}}]\,.\mathbf{1}_k)^2 \qquad (4)$$

where k in the dimension of the input vector $\mathbf{x}$, $\mathbf{1}_k$ is a k-dimensional vector of ones and $\|.\|_2$ is the L_2 norm.

3. EXPERIMENTAL SETUP

3.1. Datasets used for the study

We primarily use a Hindi dataset [19] in the source domain. We also use a Telugu dataset [20] in the ablation studies. Both these datasets are the same as the ones used in multilingual and code-switching ASR challenge, Interspeech-2021. The details of both these datasets are available in [21]. Both the speech data and the corresponding transcriptions are available. Hindi and Telugu audio files have sampling frequencies of 8 and 16 kHz, respectively. Both have 16-bit encoding. Telugu audio is downsampled to 8 kHz in our experiments. We randomly select 15,000 utterances ($\approx$ 15 hours) from their train sets for training the domain-independent acoustic models. The senone labels required for training the acoustic model are obtained from the alignments generated by a HMM-GMM system [22] trained using Kaldi [23]. We also use a random selection of 1000 utterances from the test set to validate the domain independence of learned features. We refer to this set as *dev*.

The Sanskrit dataset used in the target domain has 3395 utterances with 16 kHz sampling frequency and 16-bit encoding. The data is randomly divided into two sets - train and test, with approximately 5.5 hours (2837 utterances) in the train set and 1 hour (558 utterances) in the test set. The train set is used for domain training, and the test set is used for inference. The data is downsampled to 8 kHz before its use in all our experiments. The text corpus for building the Sanskrit language models makes use of the *wiki* Sanskrit data dump [24] and data from several Sanskrit websites. The extracted text is cleaned to remove unwanted characters and pre-processed to restrict the graphemes to the Devanagari Unicode symbols.

3.2. Details of feature extraction

We use 40-dimensional filterbank features together with their delta and acceleration coefficients. Cepstral mean variance normalisation is performed at the utterance level. The features are spliced with a left and right context of 5 frames each. Thus the acoustic feature vector at the input of the DNN has dimensions of 1320 (40 x 3 x 11). Feature extraction is performed using Kaldi [23].

3.3. Training of the GRL model

The feature extractor (G_f) has six hidden layers with 1024 nodes in each layer. The input to the G_f is a 1320-dimensional acoustic feature vector, and the output is a 1024-dimensional feature **f**. The feature vector **f** is forwarded to both the senone classifier G_y and the domain classifier G_d. The senone classifier has two hidden layers, each with 1024 nodes and an output layer with 3080 nodes (equal to the number of senones in the Hindi training data). Domain classifier has a hidden layer with 256 nodes and an output layer with two nodes corresponding to the source and target domains. All the hidden layers are followed by batch normalization and ReLU activation. The logarithm of the softmax is computed at the output of both domain and senone classifiers. All the parameters (θ_f, θ_y and θ_d) are updated during the training with Hindi utterances. Only θ_f and θ_d are updated during the training with unlabelled Sanskrit utterances.

Negative log-likelihood loss is used for training. The models are trained using stochastic gradient descent with momentum [25]. We use a batch size of 32 and an initial learning rate of 0.01. The learning rate is scaled by a factor of 0.95 after every 20000 steps. Training is performed for 20 epochs. The same number of frames from the source and target domains are used for training at every epoch. The domain adaptation factor α is gradually changed from 0 to 1 using the approach in [8].

3.4. Training of the DSN model

Acoustic frames from the source and target domains, both of dimension 1320, are Input to the DSN. Private encoders for the source and target have four hidden layers with 512 nodes in each layer. The shared encoder has six hidden layers with 1024 nodes each. The senone and domain classifiers have the same architecture as the GRL model. All the hidden layers are followed by batch normalization and ReLU activation. The shared decoder has three hidden layers and an output layer with 1320 nodes.

The hyperparameters β, γ, and δ are chosen as 0.25, 0.075, and 0.1, respectively. In order to promote the learning of the senone classifier in the initial phase of training, domain adversarial similarity losses are activated only after 10000 steps. The rest of the training process is the same as in the GRL model.

3.5. Decoding

During the inference stage, only the output of the senone classifier is considered. The pre-softmax output of the senone classifier is normalized using the log probability of priors. To find the most probable word sequence, we use weighted finite-state transducers (WFST) [26] based decoding.

The vocabulary of Sanskrit is distinct from that of Hindi. So the FSTs for grammar (G) and lexicon (L) are built using the text corpus collected in Sanskrit (target domain). Pronunciation dictionary for building the L-FST uses the SLP1-M grapheme to phoneme (G2P) mapping scheme in Sanskrit as described in [18]. This is different from the Hindi G2P scheme in aspects like

schwa deletion and pronunciation of visargas (◌ः) [15]. HMM (H) and context-dependency (C) FSTs are created using the HMMs learned from the source domain data. These four FSTs are composed to form a single HCLG graph, which maps the senones directly to the words in the target domain.

4. RESULTS

We decode the Sanskrit test set of 558 utterances with the adversarially trained GRL and DSN architectures. These models use both the labeled Hindi data and the unlabelled Sanskrit data for training. In order to benchmark the performance of these UDA models, we decode the utterances using a simple DNN model trained only with the Hindi speech data. In this model, the domain classifier is not part of the network architecture. We also compare our results with a DNN model trained in multi-task (MT) learning setup, training the whole network to minimize both the senone and domain classification objectives. This network has the same architecture as the network in Fig. 1, except that it does not have the GRL. This model also makes use of both the Hindi and Sanskrit data. All our experiments use bi-gram word-level language models.

Table 1: WER on the Sanskrit-test set. *wiki* data dump and web-crawled data are used to prepare the text corpus for building L and G FSTs (≈ 436840 words).

Model	Source	Target	WER
DNN	Hindi	-	24.58%
MT	Hindi	Sanskrit	21.43%
GRL	Hindi	Sanskrit	17.87%
DSN	Hindi	Sanskrit	17.26%

The performance measure used to evaluate these models is word error rate (WER). GRL approach gives an absolute improvement of 6.71% in WER over the baseline DNN model when the language model is trained on a large text corpus in Sanskrit. DSN provides an absolute improvement of 7.32%. Both the UDA models have nearly similar performance, and they beat the multi-task learning model by more than 3.5%. We have also tried to fine-tune the UDA models using the senone labels of the Sanskrit train set computed from the DNN models trained on Hindi, but they did not improve the performance of the models.

4.1. Ablation studies

4.1.1. Effect of the amount of unlabelled training data from the target domain

Next, we address the question of the amount of unlabelled data required for proper adaptation. We split the training set in the target domain into six sets with 0.5, 1.5, 2.5, 3.5, 4.5, and 5.5 hours of data and train the UDA models using them. The entire source domain data is used for training. The performance of these models in terms of WER is shown in Fig. 3. The performance of both the models improves as the amount of unlabelled training data increases but nearly saturates after about 2.5 hours of data in the target domain (Sanskrit).

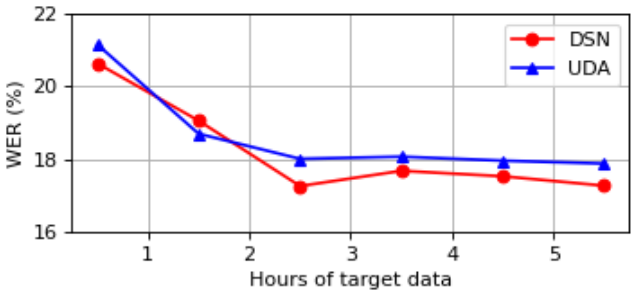

Fig. 3: Effect of the amount of unlabelled training data from the target domain on the WER.

4.1.2. Domain independence of features

We also visualize the features at the output of the feature extractor (shared encoder for DSN) for our models. We collect an equal number of frames with the same senone label (based on the HMM-GMM alignment) from the Hindi-dev and Sanskrit-test sets and plot the vectors at the output of the feature extractor (or shared encoder) using t-SNE [27]. The results are shown in Fig. 4. Compared to Figs.4(a) and 4(b), the features in Figs. 4(c) and 4(d) are more intermingled indicating the domain independence of features. The domain-discriminative power is higher for the features from MT models as seen from Fig. 4(b).

To verify the extent of domain independence achieved by the model, we compute the frame-level domain accuracy on the Hindi-dev and the Sanskrit-test sets. The results are listed in Table 2. It can be seen that the features from UDA models are much more domain-independent compared to the MT models.

Table 2: Frame-level domain accuracy of the UDA models computed on the Sanskrit-test and Hindi-dev sets.

	Domain Accuracy	
Model	Sanskrit-test	Hindi-dev
MT	91.86%	76.31%
GRL	63.77%	44.94%
DSN	63.21%	52.18%

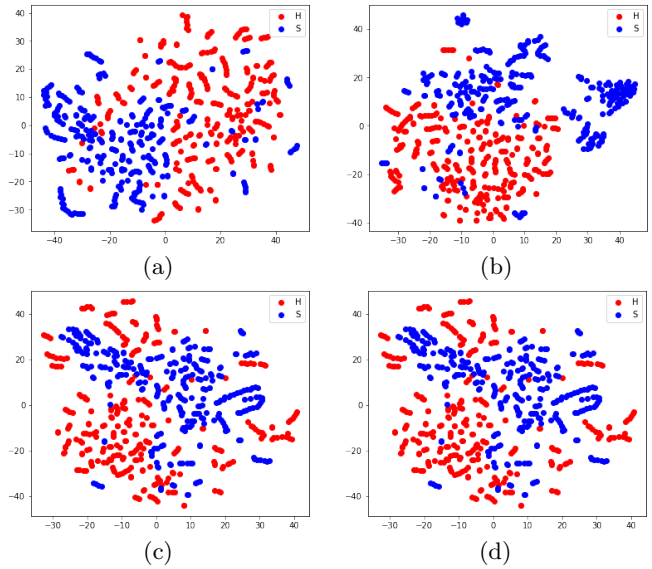

Fig. 4: 2-D t-SNE plots of features at the output of feature extractor/shared encoder in (a) baseline DNN (trained only with Hindi data), (b) MT, (c) GRL, and (d) DSN models for frames with senone label 3009. *H* and *S* denote the frames obtained from the Hindi-dev set (source domain) and Sanskrit-test set (target domain), respectively.

4.1.3. Effect of the loss functions in DSN

Next, we experiment with the constituents of the loss function in DSN. We train four models: (i) with all the loss functions in DSN, (ii) without the difference loss ($\gamma = 0$), (iii) without the similarity loss ($\beta = 0$), and (iv) with the reconstruction loss computed as SIMSE. The results are listed in Table 3. The performance degrades slightly in the absence of difference loss which tries to enhance the orthogonality between the private and shared components. There is considerable degradation in the performance in the absence of similarity loss. The results are still better than the baseline DNN and MT models (refer WER in Table 1) indicating the usefulness of private and shared component decomposition in DSN. The model using SIMSE as the reconstruction loss performs inferior to that of the one using MSE.

Table 3: Effect of the different constituents of the loss function on the performance of the DSN.

Loss functions	**WER**
All terms included	17.26%
$L_{diff} = 0$ ($\gamma = 0$)	18.10%
$L_{sim} = 0$ ($\beta = 0$)	20.37%
L_{recon} = SIMSE	18.00%

4.1.4. Source domain language selection

Though both Hindi and Sanskrit are written in Devanagari, they have a difference in pronunciation, as pointed out in section 1. However, [28] suggests that Telugu and Malayalam are the closest languages to Sanskrit in terms of pronunciation, vocabulary, and grammar. Moreover, the schwa is retained in Dravidian languages like Telugu and Malayalam, just like Sanskrit. Since suitable datasets are available in Telugu, we repeat the experiment with Telugu in the source domain, hoping for better acoustic and pronunciation models in Sanskrit. The results are shown in Table 4. All the models improve, as the phone HMM models learned from Telugu better match Sanskrit. Here also, the performance of UDA models is better than the DNN and MT models. The UDA approaches improve by around 3.5-4.5% compared to adaptation from Hindi.

Table 4: WER on the Sanskrit-test set when trained with Telugu as the source domain language.

Model	**Source**	**Target**	**WER**
DNN	Telugu	-	17.65%
MT	Telugu	Sanskrit	14.71%
GRL	Telugu	Sanskrit	13.09%
DSN	Telugu	Sanskrit	13.72%

5. CONCLUSIONS

In this work, we propose UDA as an option to tackle the scarcity of data in low-resource languages which share a common acoustic space with a high-resource language. We experiment with Hindi as the source domain language and Sanskrit as the target domain language. GRL and DSN models improve the WER by 6.71% and 7.32%, respectively, compared to a baseline DNN model trained only on Hindi. The models perform better than the multi-task learning framework. Proper selection of source domain language (Telugu in our case) further improves the results. The results indicate that UDA can provide a faster way of building ASR systems in low-resource languages, reducing the hassle of collecting large amounts of annotated training data if a suitable high-resource language is available.

6. ACKNOWLEDGMENT

We thank Science and Engineering Research Board, Government of India for partially funding this research through the IMPRINT2 project, IMP/2018/000504.

7. REFERENCES

[1] G. Adda et al., "Breaking the unwritten language barrier: The BULB project," *Procedia Computer Science*, vol. 81, pp. 8–14, 2016.

[2] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (VTLP) improves speech recognition," *International Conference on Machine Learning (ICML) Workshop on deep learning for audio, speech, and language processing*, 2013.

[3] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 9, pp. 1469–1477, 2015.

[4] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2015.

[5] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[6] A. Ragni, K. Knill, S. Rath, and M.J.F. Gales, "Data augmentation for low resource languages," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, pp. 810–814, Jan 2014.

[7] D. S. Park, W. Chan, Y. Zhang, C. C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, Sep 2019.

[8] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *32nd International Conference on Machine Learning, ICML 2015*, vol. 2, pp. 1180–1189, 2015.

[9] A. Tripathi, A. Mohan, S. Anand, and M. Singh, "Adversarial learning of raw speech features for domain invariant speech recognition," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2018-April, pp. 5959–5963, 2018.

[10] M. Abdel Wahab and C. Busso, "Domain adversarial for acoustic emotion recognition," *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 26, no. 12, pp. 2423–2435, 2018.

[11] S. Sun, B. Zhang, L. Xie, and Y. Zhang, "An unsupervised deep domain adaptation approach for robust speech recognition," *Neurocomputing*, vol. 257, pp. 79–87, 2017.

[12] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," *Advances in Neural Information Processing Systems*, pp. 343–351, 2016.

[13] Z. Meng, Z. Chen, V. Mazalov, J. Li, and Y. Gong, "Unsupervised adaptation with domain separation networks for robust speech recognition," *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017 - Proceedings*, vol. 2018-January, pp. 214–221, 2018.

[14] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[15] Anoop C. S. and A. G. Ramakrishnan, "Automatic speech recognition for Sanskrit," *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*, vol. 1, pp. 1146–1151, 2019.

[16] D. Adiga, R. Kumar, Krishna A., P. Jyothi, G. Ramakrishnan, and P. Goyal, "Automatic speech recognition in Sanskrit: A new speech corpus and modelling insights," *59th Annual Meeting of the Association for Computational Linguistics (ACL Findings)*, 2021.

[17] Anoop C. S. and A. G. Ramakrishnan, "CTC-based end-to-end ASR for the low resource Sanskrit language with spectrogram augmentation," *2021 National Conference on Communications (NCC)*, pp. 1–6, 2021.

[18] Anoop C. S. and A. G. Ramakrishnan, "Effect of different G2P schemes for Sanskrit on the performance of speech recognition models," *Proc. of 28th International Conference on Neural Information Processing (ICONIP)*, 2021.

[19] "Hindi dataset," `https://navana-tech.github.io/IS21SS-indicASRchallenge/data.html`.

[20] "Telugu dataset," `https://navana-tech.github.io/IS21SS-indicASRchallenge/data.html`, Data provided by Microsoft and SpeechOcean.com.

[21] D. Anuj et al., "Multilingual and code-switching ASR challenges for low resource Indian languages," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2021.

[22] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb 1989.

[23] D. Povey et al., "The Kaldi speech recognition toolkit," *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, Dec. 2011.

[24] "Wiki Sanskrit data dump," `https://dumps.wikimedia.org/sawiki/`.

[25] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," *30th International Conference on Machine Learning, ICML 2013*, , no. PART 3, pp. 2176–2184, 2013.

[26] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer, Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[27] L. Van Der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2625, 2008.

[28] P. Sreekumar, *Sanskrit superstratum and the development of Telugu and Malayalam: A comparative note*, International School of Dravidian Linguistics, VI Subramoniam commemoration: Studies on Dravidian, vol 1, pp. 169-180, ISBN 81-85692-60-2, 2015.