Towards improving pre-trained models for Sanskrit NLU

Natural Language Processing

Shubharup Ganguly
BSDS, IIT Madras

Under the guidance of
Prof. Pawan Goyal and Manoj Balaji
IIT Kharagpur

December, 2023

Statement of Originality

I affirm that the structure and written contents of this exposition are wholly original. There has been no instance of material taken directly from another source, except where attributed. I am committed to academic integrity as a necessity, in my current and future works.

Acknowledgments

I would first and foremost like to thank my mother, Smt. Swarupa Ganguly for raising me as well as she did despite the hardships, and for supporting me in my academic journey so far. She is the lifeline without which none of it would have been possible.

I would like to thank Manoj Balaji, for patience in introducing me to the ideas at the frontier, and trust in and encouragement of my research capabilities.

I owe no small thanks to the following professors, who are the first teachers I have had in the field of machine learning. Prof. Pawan Goyal, for the course 'Natural Language Processing'. Prof. Arun Rajkumar, for the course 'Machine Learning Techniques'. Prof. Mitesh Khapra, for the course 'Deep Learning'.

I thank the NPTEL initiative and its internship programme, for enabling valuable learning outcomes for me through its courses, and for the opportunity to produce original research.

I would like to thank the professors of the IIT Madras BS Degree programme. I have gained an incredible sum of knowledge and skills from the program, and I can honestly say that the IITMBS experience so far has been a journey in academic and professional growth, as much as also a catalyst in my personal growth.

I would like to thank all my former professors, at Jadavpur University and RKMRC Narendrapur. "There is no such thing as a 'self-made man' ", as the saying goes[1], and it is my education which has made me. Thank you.

[1] "There is no such thing as a self-made man. We are made up of thousands of others. Every one who has ever done a kind deed for us, or spoken one word of encouragement to us, has entered into the make-up of our character and of our thoughts, as well as our success."

Contents

1 Introduction

Disambiguation

Unless explicitly mentioned otherwise, throughout this document 'Sanskrit' refers to *classical* Sanskrit rather than Vedic Sanskrit. Although related to one another as ancestor and descendent, they represent distinct challenge areas as of the present day.

1.1 The motivation for a Sanskrit LLM

This quotation [1] is a good start:

"Its importance is plain: Sanskrit was **once the most influential literary language in India**, and texts written in the language could be understood by millions of people throughout the South Asian world. These texts contain **profound meditations on every point on the spectrum of human concern**: existence, reality, God, love, duty, marriage, war, sex, death, violence, laughter, beauty, perception, nature, anatomy, urbanity, ritual, desire, food, purpose, meaning, and language, among hundreds of others. Moreover, Sanskrit texts are the **repository of non-modern modes of thought**, and they present **distinct conceptions of the world** that are often at odds with the understanding we have today. By learning how people used to think, we better understand both ourselves and the world we have inherited."

Indeed, Sanskrit is the language of an extensive body of treatises in wide and varied domains, some of which are:

- Linguistics: Aṣṭādhyāyī of Panini, a system of generative grammar proposed some millennia before the Chomskyan syntactic paradigm, "whose level of detail and theoretical sophistication has not been equaled to this day" [2]
- Logic: Tattvachintamani of Gaṅgeśa, a treatise on formal proof paradigms and rhetorics
- Poetics: Chandaḥśāstra by Pingala
- Statecraft: Arthashastra by Kautilya
- Medicine: Caraka-Saṃhitā by Charaka, Suśrutasaṃhitā by Sushruta
- Religion and Spirituality: Śrīmadbhagavadgītā by Veda-Vyasa

(This author must apologize in advance for the error in omission of other prominent titles, in particular the vast literary body of prose, poetry, mythology, etc. to which I have had

[1] *Introductory foreward to www.learnsanskrit.org* (n.d.)
[2] Jamison (2008)

lesser exposure.)

With Yoga and Ayurveda becoming global phenomena since the 20th century, there is a **rising interest in the indigenous knowledge systems of India**, to which currently knowledge of Sanskrit is the gatekeeper. Successful work in NLP and AI to make this vast reserve of knowledge more accessible for a general audience would benefit generations to come, by way also of preserving an ancient heritage.

1.2 Challenges for Sanskrit NLP

1.2.1 Linguistic Productivity

Sanskrit is credited to be a very productive language, which adds to the challenge of Sanskrit Natural Language Understanding because many words are hapax legomena i.e. seen only once or a scarce few times, and although they may be created from previously seen constituents (like nouns, verbs and affixes) they may have substantially different meaning from the sum of the meanings of the parts.

Panini does not stipulate an upper-bound on the length of derivation and compounding. Kulkarni and Shukl (2009) write that:

"Though compounds of 2 or 3 words are more frequent, compounds involving more than 3 constituent words with some compounds even running through pages is very common in Sanskrit literature."

1.2.2 Sandhi and Polysemy

Sanskrit has a long history of oral knowledge transmission. There is a quotation [3] which perhaps captures the motivation:

"If your knowledge is just in a book,
Or your money in somebody's care,
You will find in the moment of action,
Neither are anymore there."

To facilitate ease of speaking and remembering, adjacent words were often merged, in accordance with certain phonological rules occuring at the boundaries, to form one long unbroken sentential utterance. A considerable volume of Sanskrit text is found in this form.

Sandhi-spliting is an additional pre-processing step that isn't relevant for languages like English which have the orthographic convention of splitting a sentence into its constituent words with whitespaces.

Sandhi-splitting is a challenge, because a single sentence can have several possible splits with distinct compositional meanings.

Furthermore, Sanskrit words are often polysemous i.e. one word can have several possible meanings or interpretations (i.e. a one to many mapping from symbol to meaning).

1.2.3 Scarcity of data

Because of the historical exclusivity and the emphasis on oral transmission, textual data for Sanskrit is limited to begin with.

[3] cited in Wujastyk (1981)

Because of the separation of time, a smaller sample of what was produced survives to us.

And because of the limited numbers of present day speakers, new data is not being created in significant amounts (compared to some languages with explosive growth, such as English).

There is ongoing effort at digitalizing Sanskrit documents, and at automated digitalization with OCR. (see Avadesh and Goyal 2018 for an example)

Following from these efforts, larger digital corpora for machine intelligence tasks are also emerging e.g. the *itihasa* dataset released by Aralikatte et al. (2021).

With a moderate sized corpus, it is still possible to create a pre-trained language model. But there is some evidence to suggest the pre-training methods employed for high-resource languages may not carry over, as is, to producing good quality embeddings in low-resource settings with limited data. Alabi et al. (2020) present such a case study for two African languages Yorùbá and Twi, where the authors find that embeddings from generic models are outperformed by language-specific processing with a curated corpora. Konovalov and Tumunbayarova (2018) report that for the relatively small corpus available for the Buryat language, traditional count-based methods outperform neural-based methods.

There is also the phenomena of emergent behavior in Large Language Models – the capabilities of a language model do not seem to scale linearly with data and number of parameters; instead, there is a 'thresholding' effect, such that once the model reaches a critical mass of data and number of parameters (and hardware availability for the computation), the model's performance scales up dramatically i.e. at a rate not predictable from the preceding trend. (See Wei et al. 2022 for a review.)

1.2.4 Skill gap

Kulkarni and Shukl (2009) write:

"Though well defined rules for Sanskrit morphology exist in Astadhyayi, for a typical computational linguist without any knowledge of Sanskrit, it is difficult to build a system incorporating these rules. At the same time, it is rare to find a person who understands the Astadhyayi well and has also a good knowledge of computers for its implementation."

This situation may be changing for the better in current times.

2 Pre-trained Language Models

2.1 Representing words – Word Embeddings

One-hot encoding This is the most straightforward way of representing words with numbers. Each word in the vocabulary is assigned a vector of length $|V|$ (the size of the vocabulary, i.e. number of distinct words). Each such vector is 'on' or 'hot' i.e. has a 1 in one exactly one position, and 0s in all others. Each of the $|V|$ words in the vocabulary have such a distinct binary vector associated with it.

One hot encodings are high-dimensional and sparse, their storage requirements scale up rapidly with the size of the vocabulary, and they do not capture semantic similarities because all words are equidistant from and orthogonal to all other words.

Word embeddings A word embedding is a vector of numbers, which can be thought of as the meaning content of a quantal word which is understandable to machines.

Good embeddings should capture some intuitive notions about words, such as:

- pairwise distance: 'cat' should be closer to 'dog', than 'cat' is to 'aeroplane'.
- proportionality: queen is to king, as headmistress is to what? (headmaster)

2.1.1 Word embeddings pre-2013 (before deep learning)

Word embeddings before deep learning can largely be said to be *count-based*. This count-based approach is essentially based on the idea of collocational meaning in distributional semantics, succinctly encapsulated with a popularly circulated quotation by British linguist J.R. Firth:

> "You shall know a word by the company it keeps." (Firth 1957)

Co-occurrence matrix and PPMI matrix The co-occurence matrix is a tabulation of the co-occurrence statistics of each word with every other word in the vocabulary. (More specifically, this is the Term-Term co-occurence matrix, and there are other kinds of co-occurrence matrices such as Term-Document and Document-Document.)

The co-occurrence matrix is often converted to a Pointwise Mutual Information (PMI) matrix. One motivation for this is that this reduces the influence of very frequent words, as we want it to be such that the co-occurrence of a word with some very frequent word should not be taken to be as being very informative.

Singular Value Decomposition (SVD) The co-occurrence matrix or PPMI matrix is then factorized using a matrix factorization technique called Singular Value Decomposition (SVD).

SVD factorizes the co-occurrence matrix into a product of three matrices: $A = U\Sigma V^T$. This factorization then lets us get the best k-rank approximation of the co-occurrence matrix, which is also called the truncated SVD of the original co-occurrence matrix A. E.g. if we want k-dimensional word embeddings, then we take the first k columns of U, call it U_k, and then the k-dimensional row vectors of U_k are the word embeddings.

A Term-Term co-occurrence matrix is symmetric, and the matrices U and V are the same. But in a Term-Document co-occurence matrix (usually not symmetric), the matrices U and V are not the same. The rows of U_k are still the word embeddings. Taking the first k rows of V^T and calling it V_k^T, the k-dimensional columns of V_k^T are the embeddings of the documents as *contexts* aka context embeddings.

With the advent of deep learning methods, purely count-based method went out of fashion. Baroni, Dinu, and Kruszewski (2014) demonstrate the superiority of embeddings from SGNS in several evaluation tasks.

Levy and Goldberg (2014) showed Skip-Gram with Negative Sampling (SGNS) to be implicitly doing the factorization of a PMI co-occurrence matrix. Levy, Goldberg, and Dagan (2015) then went on to show that with some modifications SVD performed at least as well as SGNS on similarity tasks, but not on analogy tasks.

2.1.2 Word embeddings with deep learning

Continuous Bag of Words (CBOW) A Continuous Bag of Words (CBOW) model attempts to predict the upcoming word given the preceding words (i.e. the context leading up to it), and adjusts its internal parameters according to the feedback from the prediction error (cross entropy loss).

This is called *conditioning* on the context.

Skip-gram While CBOW predicts a word given the context, Skip-gram attempts to predict the context words given an input word. Skip-Gram is often done in conjunction with Negative Sampling, a kind of data augmentation technique where for every (word, context) pair (positive example) which actually occurs in the data, *non-occurring* (word, context) pairs (negative examples) are artificially added to the data using some random sampling. Skip-Gram with negative sampling is often denoted as SGNS.

CBOW and Skip-gram were both introduced by Mikolov et al. (2013) in their Word2Vec package, and each has its distinct advantages. CBOW is faster to train than Skip-gram. CBOW better represents more frequent words, while Skip-gram better represents rarer words. CBOW learns better syntactic relationships, while Skip-gram learns better semantic relationships.

Global Vectors (GloVe) GloVe embeddings are a hybrid of prediction-based and count-based methods of creating embeddings.

The key idea behind GloVe is to leverage global information about word co-occurrence patterns across the entire dataset.

Like count-based methods, it first constructs a word co-occurrence matrix.

The objective of GloVe then is to learn word vectors v_i, v_j such that the dot product of these vectors $< v_i, v_j >$ approximates the logarithm of seeing one word in the context of the other word, for both words – $< v_i, v_j > \approx logP(i|j)$ and $< v_i, v_j > \approx logP(j|i)$.

fastText Whereas the original CBOW, Skip-gram and GloVe produce embeddings at the word-level, fastText produces embeddings at the sub-word level.

At least one work on Indic languages (Kakwani et al. 2020) have used fastText as their choice for producing embeddings because of its ability to handle the morphological complexity that is characteristic to Indic languages, and so better handle out-of-distribution words which are actually previously unseen re-combinations of seen subwords.

ElMo One major issue with all the embedding approaches discussed so far (both the count-based SVD factorization, and the four prediction-based methods listed above) is that they produce a single embedding for each word, thus neglecting polysemy. So the word 'bank' in both sentences below would get the same embedding:

1. The bank is down the street. (referring to the building of a financial institution)

2. The banks are made up of silt deposition. (referring to a river bank)

This was first addressed in Peters et al. (2018) – their model, **E**mbeddings from **L**anguage **Mo**dels aka ELMo, was the first to produce *contextualized embeddings* – different embeddings for the same word according to the meaning of their occurrence in a specific context.

2.2 Transformers

2.2.1 Encoder-Decoder Architecture

Encoder-Decoder is a kind of architecture for deep neural networks where the task of 'understanding' an input (encoding) is decoupled from producing an output based on the input (decoding). The encoder takes the input and encodes it into an input representation. The decoder takes this encoded representation as input, and produces the final output. One intuitive example of this is a cross-modal application like Image Captioning, where the Encoder might be a CNN, and the Decoder may be an RNN (or variant, such as LSTM).

Sequence-to-Sequence (Seq2Seq) models are a kind of encoder-decoder model architecture, using an RNN or variant for both the encoder and decoder, which can take a sequence as input to produce a sequence as output. Certain machine intelligence tasks can be cast as a Seq2Seq task e.g.

1. Machine translation: an input sentence is a sequence of words in the source language, whereas the output sentence would be a sequence of words in the target language

2. Text Summarization The original document is a sequence of words, the produced summary is also a sequence of words.

2.2.2 Attention

Attention is an architectural component introduced in Bahdanau, Cho, and Bengio (2014) for Seq2Seq which enables the model to selectively focus more on inputs at some timesteps, when producing the output at a timestep. At every decoder timestep, as the decoder decides its output at that timestep, the decoder receives not just the final encoder hidden state, but *all* hidden states (i.e. the hidden states at all encoder timesteps), and uses a similarity mechanism to weigh these states by importance, before weighted-averaging them to make up the final decoder input at the timestep.

(The encoder hidden state at a particular encoder timestep is the representation of the input word at that timestep, in the context set by the words leading up to it.)

- In Seq2Seq models before Attention, the decoder only used the encoded representation of the entire input i.e. only the final hidden state.
- With Attention, the decoder additionally uses the encoder hidden states for every timestep (and not just the final hidden state). The decoder weighs these hidden states with softmaxed attention scores.

That the decoder could now receive all hidden states, weighted by importance, fixed the limitations of RNNs and their variants in working with (understanding or producing) longer sequences.

2.2.3 Transformers

But RNNs and variants were still slow to train, because the recurrence in them was inherently serial in nature, not lending itself to parallelization.

The need for RNNs, for either the encoder and/or the decoder, was altogether eliminated with the introduction of the Transformer architecture (Vaswani et al. 2017).

Vaswani et al. (2017) introduced some innovations to the standard architecture of an encoder-decoder model with attention:

1. **Self-attention:** Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence.

2. **Key/Value/Query paradigm:** The key/value/query concept is analogous to retrieval systems. For example, to search for a video on Youtube, the search engine maps the search query against the keys (descriptive representations of candidate videos) in the database, then presents the best matches (values).

   The attention operation can be thought of as a retrieval process as well.

   This abstraction is motivated by the fact that an item in the sequence (*the embedding of* a word in a sentence) plays three distinct roles during the attention mechanism process:

   - One, for the attention weights on itself and the other words when its own representation is being computed, it is the Query;
   - Two, for the attention weight on it when the representation of some other word is being computed, it is the Key;
   - Three, during the attention-weighted aggregation for the computation of either its own representation or some other word's, it is the Value.

3. **Multi-Head Attention:** The self-attention mechanism uses a set of three matrices – W^Q, W^K and W^V – to project a word (embedding) as Query, Key and Value vectors respectively. (These matrices are learnable as part of the optimization algorithm.)

   One set of these matrices (W^Q, W^K, W^V) is called an attention head.

   Each attention head is its own attention mechanism – from the sentence of words i.e. set of inputs, it produces a set of encoded inputs, **incorporating knowledge of some particular relation(s) between the words**, including but not limited to grammatical relations.

There is a principle of specialization and division of labor at work here, i.e. **different attention heads take on different responsibilities i.e. the relations** between the items of the input sequence **which they are responsible for encoding** (one word can be related to another word with possibly more than one relations); and so the set of encoded inputs is different for different attention heads, for the same input sequence.

These different encodings for a word are all concatenated, and brought down to the original dimension length by a matrix multiplication.

The major motivation for multiple attention heads is that, the final sequence of vectors (after that last dimensionality reduction) are 'rich' embeddings of the original words – rich in that they've captured multiple inter-relationships between the input words.

2.3 Transfer Learning

2.3.1 Task transfer

What are some tasks, which if you get good at, there is the side-effect of simultaneous improvement at a broad spectrum of other related tasks?

Task transfer is the carryover of proficiency in one task acquired by practice or training, to one or more other tasks.

Think of performance aptitudes which have a specific kind of correlation: an improvement in one has the side-effect of improvement in the other (but not necessarily equally in both directions).

The idea of task transfer is quite broad, and extends to human cognitive and physical domains. In the domain of physical culture, strength improvement in compound movements (multi-joint movements e.g. the "bench press", which is executed with simultaneous pectoral adduction and tricep extension) often carries over to isolated movements (movements over a single joint e.g. the "pec fly", requiring only pectoral adduction).

Now for a more proximal example, it is recognized in hobbyist language-learning communities that there is carry over of linguistic competence from learning some target language, to then learning a language mutually intelligible with the former e.g. from Hindi to Urdu, Spanish to Portugese, and vice versa.

Although the ground-up implementation of this process in machines looks very different from our experience of human-level processes, it turns out something very similar can be worked into machine intelligence.

That is, it is possible to first train a model on some general task, such that this general competence model performs much better on many later tasks even with comparatively light instruction on those later tasks.

In the machine intelligence literature, the first stage, of training on a general task, is called *pre-training*. The next task-specific instructing of a pre-trained model is called *fine-tuning* the model for that task.

2.3.2 Unsupervised Pretraining

The name unsupervised pre-training is a slight misnomer; the process is more accurately called **self-supervised** – the raw training data itself provides the supervision signal.

Contrary to the standard conception of unsupervised learning, there *are* targets (viz. 'ground truth' i.e. exemplary correct class predictions) that the model is using for its goal-directed learning process, of updating its internal parameters so as to minimize the loss (error) of its predictions with respect to the correct predictions (targets).

But these target labels are created in an automatic fashion, without the need for the kind of manual intervention traditionally needed to create data for supervised learning e.g. annotations. This latter reason is why it is unsupervised – semi-supervised is a specific subset of unsupervised.

By way of an analogy with a classic machine learning technique, pre-training is to task training, what K-means++ is to K-means: the specific matter of difference is of better initialization. The outcome of pre-training is essentially better initialization of parameters for subsequent fine-tuning for a downstream task, where the latter fine-tuning is the supervised training for a specific task like Word Sense Disambiguation (WSD) and Named Entity Recognition (NER).

One intuition for why pre-training works i.e. produces task transfer capabilities, is as follows: in a stochastic (random, blind) search in a high dimensional search space of parameters, over a loss surface that is highly non-convex (multiple ponds and oceans instead of nicely cup-shaped), pre-training goes a step towards improving the odds that the algorithm (backpropagation, with some variant of gradient descent like Adam) settles down in a 'good' initialization landscape: good being a region with many local optima in the neighborhood, with the optima for the downstream fine-tuning task being one of those neighborhood optima. Li et al. (2018) present experimental support in favor of such a hypothesis.

2.3.3 Pretraining objectives

The task on which the model is called the 'pre-training objective'.

For a recent survey of language representation modelling and pre-training objectives for it , refer to Schomacker and Tropmann-Frick (2021).

In the next two sections, I overview two of the most widely known pre-trained models of recent times, and describe their pre-training methodology.

2.3.4 BERT (Google 2018)

BERT was pre-trained on two tasks viz. (1) Masked Language Modelling (MLM) aka the cloze task and (2) Next Sentence Prediction (NSP).

The cloze task The MLM objective is also called the **cloze task**.

In a training sample, a certain percentage of the words were randomly selected and replaced with a special [MASK] token.

The model was then trained to predict the original identity of the masked words.

This necessitated the model to understand the surrounding context around the [MASK] well-enough to be able to rely on it to accurately predict the masked / replaced word.

The NSP task BERT's Next Sentence Prediction (NSP) task trained the model to predict whether a pair of given sentences in a sequence were consecutive or not. This helped it learn contextual dependencies between sentences during pretraining.

BERT was trained on pairs of sentences, where each pair consists of two consecutive sentences from a larger text corpus. The input sentences were tokenized using WordPiece tok-

enization, and converted to embeddings. Then one of the sentences was randomly chosen to be masked. The model was then trained to predict whether the other (unmasked) sentence in the pair was the immediate next sentence following the first one or it was not. The training signal was provided by a binary classification task, where the model predicted either "IsNext" or "NotNext".

2.3.5 GPT-1 (OpenAI 2018)

Whereas BERT was trained on the MLM and NSP tasks, GPT-1 was trained with (only) the autoregressive language modelling task.

The input text was tokenized using Byte Pair Encoding (BPE). Each token was then represented as an embedding vector. The model was fed a context window of preceding words, and the task was to predict the probability distribution of the next word in the sequence. The training objective was to maximize the likelihood of observing the actual next word given the preceeding words. This was done by minimizing the cross-entropy loss between the predicted probability distribution, and the actual probability distribution of the next word (the latter being a one-hot vector).

After training, the model could generate text autoregressively by sampling from its predicted probability distribution for each word, conditioning on the previously generated context.

3 Sanskrit Computational Linguistics

3.1 Panini's Astadhyayi

For a considerable period, Sanskrit was an orally transmitted language.

Around 500BC, Panini, a scholar of Sanskrit grammar, produced the *Astadhyayi*, a 'snapshot' of the language of the cultured and intellectually cultivated strata of society at his time of life viz. of the grammatical regularities in it. The work may be considered both descriptive and prescriptive. For long after his decease, the *Astadhyayi* was considered as the standard reference for Sanskrit grammar. So what was composed as a descriptive grammar became a prescriptive grammar learning the language. All subsequent Sanskrit follows, or attempts to follow, the rules of Panini.

The composition of the *Astadhyayi* ushered in the era of what is known as *classical* Sanskrit, in contrast with its predecessor *Vedic* Sanskrit. (Only the language of the great epics, the *Mahabharata* and *Ramayana*, deviate somewhat from the Paninian norm, and their language is therefore sometimes distinguished as Epic Sanskrit.)

The reader who may be interested in learning more about Panini's grammatical system is suggested to refer to Kiparsky (2009) and Joshi (2009).

3.2 Some features of the Sanskrit language

In the full interest of disclosure, I must state that considerable portions of the following summary are taken directly from Jamison (2008), and I would like to thank the author for her work, and for creating such a ready reference for a layperson to Sanskrit. The author also provides a Reading List at the end for further inquiry.

- **Sandhi:** The surface form of any linguistic string is subject to phonological rules of combination (sandhi or "putting together"). In other words, phenomena of the English 'gonna' (from 'going' + 'to') type apply to any two words in contact within a sentence, and even between sentences in a discourse.

  Much of Sanskrit text is originally available in this sandhied form. This makes Sanskrit Word Segmentation (SWS), or 'sandhi-splitting' simply put, a necessary pre-processing step for Sanskrit NLP and NLU.

  Sandhi-splitting is a challenge, because a single sentence can have several possible splits with distinct compositional meanings.

- **Free Word Order:** Sanskrit has relatively free word order. That is, although there is a canonical word order (Subject-Object-Verb or SOV) which is followed in ordinary prose,

the order of words in a sentence can be scrambled without affecting the meaning of the sentence. This liberty is amply exercised in poetic and artful prose, for expressive and discourse purposes.

- **Rich morphology and case system:** The free word order of Sanskrit is afforded by its powerful morphological system and overt case marking. Because of its elaborate morphology many traditionally "syntactic" phenomena take place on the level of morphosyntax in Sanskrit. The case system allows the syntactic roles of nominals to be encoded without recourse to rigid word order or obligatory adpositions.

- **Word formation:** The basis of Sanskrit morphology are the root words, indivisible morphemes bearing lexical meaning. Through a vowel-gradation process and through the addition of affixes, verbal and nominal stems are derived from this root. The grammatical (syntactic) identity of a stem in sentential context is then fixed by the addition of an ending.

  So the three major formal elements of the morphology are (i) root, (ii) affix, and (iii) ending;

  and they are roughly responsible for

  (i) lexical meaning, (ii) derivation, and (iii) inflection, respectively.

  A (non-compound) word ordinarily contains only one root and one ending, but may have a theoretically unlimited number of affixes. Both ending and affix may also be represented as zero. The canonical structure of a Sanskrit word is thus

  $Root - \text{Affix}_{0-n} - \text{Ending}_{0-1}$

  Sanskrit morphology is conveniently divided into two fundamental categories, namely nominal forms and verbal forms, formally distinguished by the types of endings they take and the grammatical categories these endings mark. Adjectives and participles derived from verbs are not formally distinct from nouns; pronouns share the same grammatical categories with nouns, though they may deviate somewhat in inflection.

- **Compounding:** Sanskrit has an extremely well-developed system of nominal compounding; verbal compounding is rare. In Classical Sanskrit, compounds of dozens of items are not infrequent, especially in philosophical texts: the compounding process comes to take the place of the independent syntactic arrangement of inflected words. (This is very similar to the situation in German, where compounding is an 'everyday practice' in discourse.)

3.3 Sanskrit NLP

This section presents an overview of the progress in some NLP tasks for Sanskrit which are especially relevant to NLU.

3.3.1 Word segmentation

Sanskrit Word Segmentation (SWS) has been studied since at least 2005 (Huet 2005).

Around 2015, deep learning approaches to the problem started emerging (Hellwig 2015).

Prior to 2023, the approach of Hellwig and Nehrdich (2018) with a character-level recurrent and convolutional neural network represented the state-of-the-art in Sanskrit Word Segmentation.

In 2023, Sandhan (2023) proposed a novel method for SWS which reports an average of 7.2 points absolute gain over the previous state of the art.

3.3.2 Morphological parsing

Krishna et al. (2020) reports the current state-of-the-art in morphological parsing for Sanskrit, which is done jointly with dependency parsing.

3.3.3 Dependency parsing

Kulkarni and Ramakrishnamacharyulu (2013) write that –

> The parse of positional languages such as English are well expressed by constituency structure while languages like Sanskrit which are morphologically rich and to a large extent free word order are better represented by a dependency tree.

Krishna et al. (2020) reports the current state-of-the-art in dependency parsing for Sanskrit, both in standalone (with gold standard morphological tags) and with joint morphological parsing.

3.3.4 Distributional semantics

Kumar et al. (2020) is a massive release of embeddings for Sanskrit (and other Indian languages) from different methods such as BERT, ELMo.

Bhatnagar, Lonka, Kunal, et al. (2023) is one of the early published attempts to use a pre-trained model for Sanskrit on a downstream task viz. extractive text summarization.

4 Suggested future directions

4.1 Better pre-training objectives

The cloze test was proposed in 1953 by psychologist Wilson Taylor [1], who derived inspiration for the task and its name from the **Gestalt psychology concept of closure**, a general principle of human pattern recognition: that human minds are predisposed to form a complete whole, so even when some bits or parts are missing, our minds will automatically leverage some pattern recognition to fill in the gaps so as to reconstruct the whole.

Originally, the cloze task was used to compute metrics for reading comprehensibility of a body of text: lower reading comprehensibility would result in generally lower performance on the cloze task by a sample set of participants.

Then it was repurposed, "spun around", and gained much attention in the 1970s and 1980s as an exercise for teaching English as a foreign language – from measuring reading comprehensibility of a text, to building language comprehension in language learners.

Now again it is *seemingly* repurposed, in a vein rather similar to its last iteration, as a task for the pre-training of language models.

It is notable here that, it is unclear whether in the machine intelligence track there was prior awareness of the history of the cloze task in language pedagogy, or its roots in gestalt theories of psychology and human cognition. Adoption of the same name does suggest it, but it deserves emphasis that the perspective in machine learning may not have been intentionally convergent.

Nevertheless, this history itself is interesting, because it suggests avenues for interdisciplinary research and collaboration with domains not yet associated with machine intelligence:

- Research in gestalt psychology, alternative theories, and the domain of human pattern recognition in psychology [2]
- Teaching English as a Foreign Language (TEFL), Task Based Language Teaching (TBLT), Task Based Language Assessment (TBLA)

Ellis et al. (2020) and Sudharshana and Mukhopadhyay (2022) are reference material for Task Based Language Teaching (TBLT) and Task Based Language Assessment (TBLA), respectively.

After the MLM and NSP tasks of BERT, newer pre-training objectives have been proposed which are more sample-efficient and compute-efficient e.g. the Replaced Token Detection

[1] Taylor (1953)
[2] *Pattern recognition (psychology)*

task (Clark et al. 2020). This is a language-agnostic pre-training task that has been used, for example, to train BERT for Bangla by Bhattacharjee et al. (2021).

In the future, we can expect to see pre-training tasks specifically designed according to (1) the design features of the language and (2) the nature of the corpus available.

Sanskrit is intrinsically different from English in being morpho-syntactically richer. Syntactic relations are overtly marked by *karaka*, which allows for a free word order without causing ambiguity.

Drawing analogy from the history of the cloze task in English language pedagogy, the motivation should be apparent for better documentation and exploration of the language tasks used in the traditional Sanskrit teaching lineage and oral tradition.

But after perusal, there seems to be a surprising, and unfortunate lack of literature on this subject in the public domain.

So far, I have found two traditional Sanskrit teaching methods, mentioned in Wujastyk (1981) and Shukla et al. (2016). Their descriptions are taken out of the latter.

1. *Dandanvaya:* Also known as *Anvayamukhi*, where the task is to arrange all the words in prose order according to their grammatical function and syntactical relation, thus rendering the verse easy to understand.

2. *Khandanvaya:* Also known as Dialogue / Dialectic Method or Method of Questions and Answers, it resembles *Dandanvaya* in first picking out the principal sentence, but differs from it regarding the construing of remaining words of the whole sentence, which is carried on in sections (*khandas*) by framing questions on individual words or phrases. These questions are centered around the heads seeking their various modifiers.

4.2 More on Task Transfer

Benson (2016) can be used as an initial reference for the phenomena of task transfer in human language pedagogy.

As Benson writes – "Two key issues in TBLT are **(1) task selection and (2) sequencing** for both teaching and assessment."

This is in line with **'the cognition hypothesis'** in task-based language teaching, which states that **"pedagogic tasks be sequenced for learners largely on the basis of increases in their cognitive complexity so as to increasingly approximate the demands of real-world target tasks"** (quoted from Robinson 2003).

There is a surprising parallel in the machine intelligence track, by way of at least one prominent example: **BERT is first trained with MLM, then with NSP** i.e. sequentially. With the MLM task, the model learns fine-grained representations of individual words in the context of a sentence. Once the model has gained proficiency in this, the NSP task is introduced for the model to learn to understand relationships between sentences and the broader context of a document. There is a logical progression *of increasing task complexity* in terms of understanding language from the word level to the sentence level.

It is a strong supposition, that the literature and methods on task-based language teaching and assessment may contain novel insights for the field of machine intelligence, which need to be 'translated' according to the new context of applicability. So it is a desirable for the future, to

see a channel of dialogue open up between domain experts on machine intelligence, and domain experts in human pedagogy.

There are caveats for this interdisciplinary approach, however, which should necessarily be borne in mind.

The divergence between developments in NLP and advances in our understanding of psycholinguistics, stems from the reason that machines have different endowments and constraints from humans, so that the strategies (algorithms) which can be made to work well on computers *need not* have a basis in human cognition. (see Crocker 2010 for further reading)

E.g. humans are possibly endowed with an innate faculty of language, or an innate disposition for and competence at picking up human languages, the Chomskyan 'Language Acquisition Device'. But this comes together with certain 'hardware limitations' e.g. working memory constraints, the net result being that the processes employed by the human language processor may be divergent from the best possible processes on a computer.

4.3 Theories of NLU in the Sanskrit tradition

The Indian school's ideas on verbal cognition (*sabdabodha*) will probably have relevance to Sanskrit NLU at a later time, if not the whole of the larger enterprise of building human-level language understanding capabilities on computers.

Paraphrasing from Shukla et al. (2016):

"Some essential factors which help in *sabdabodha* are mutual expectancy (*akansa*), consistency (*yogyata*), proximity (*sannidhi*) and purport (*tatparya*).

The concept of *akansa* teaches a student where to look for clues for establishing relations between the words in a sentence, *yogyata* teaches which meaning to look for among the *abhidha*, *laksana* and *vyanjana*, while *sannidhi* puts some constraints on possible combinations of relations between the words, and *tatparya* helps in disambiguation of ambiguous words and understanding the whole purport of the utterance."

Madhav Deshpande's book *The Meaning of Nouns* (Deshpande 2012) is an excellent scholarly resource on the various traditions of grammar and the theory of meaning in the Sanskritic scholarly tradition.

It is voluminous in coverage, and this again may be an as-yet unexplored reservoir of inspiration for the future of Sanskrit NLP.

Bibliography

Alabi, Jesujoba et al. (2020). "Massive vs. curated embeddings for low-resourced languages: the case of Yorùbá and Twi". In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 2754–2762.

Aralikatte, Rahul et al. (2021). "Itihasa: A large-scale corpus for Sanskrit to English translation". In: *arXiv preprint arXiv:2106.03269*.

Avadesh, Meduri and Navneet Goyal (2018). "Optical character recognition for Sanskrit using convolution neural networks". In: *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*. IEEE, pp. 447–452.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski (2014). "Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 238–247.

Benson, Susan D (2016). "Task-based language teaching: An empirical study of task transfer". In: *Language Teaching Research* 20.3, pp. 341–365.

Bhatnagar, Kartik, Sampath Lonka, Jammi Kunal, et al. (2023). "San-BERT: Extractive Summarization for Sanskrit Documents using BERT and it's variants". In: *arXiv preprint arXiv:2304.01894*.

Bhattacharjee, Abhik et al. (2021). "Banglabert: Combating embedding barrier for low-resource language understanding". In: *arXiv preprint arXiv:2101.00204*.

Clark, Kevin et al. (2020). "Electra: Pre-training text encoders as discriminators rather than generators". In: *arXiv preprint arXiv:2003.10555*.

Crocker, Matthew W (2010). "Computational psycholinguistics". In: *The handbook of computational linguistics and natural language processing*, pp. 482–513.

Deshpande, Madhav M (2012). *The meaning of nouns: Semantic theory in classical and medieval India*. Vol. 13. Springer Science & Business Media.

Ellis, Rod et al. (2020). *Task-based language teaching: Theory and practice*. Cambridge University Press.

Firth, John (1957). "A synopsis of linguistic theory, 1930-1955". In: *Studies in linguistic analysis*, pp. 10–32.

Hellwig, Oliver (2015). "Using Recurrent Neural Networks for joint compound splitting and Sandhi resolution in Sanskrit". In: *4th Biennial workshop on less-resourced languages*.

Hellwig, Oliver and Sebastian Nehrdich (2018). "Sanskrit word segmentation using character-level recurrent and convolutional neural networks". In: *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2754–2763.

Huet, Gérard (2005). "A functional toolkit for morphological and phonological processing, application to a Sanskrit tagger". In: *Journal of Functional Programming* 15.4, pp. 573–614.

Introductory foreward to www.learnsanskrit.org (n.d.). URL: `https://www.learnsanskrit.org/introduction/`.

Jamison, Stephanie; W. (2008). "Sanskrit". In: *The Ancient Languages of Asia and the Americas*. Cambridge University Press, pp. 6–32.

Joshi, Shivram Dattatray (2009). "Background of the Aṣṭādhyāyı". In: *International Sanskrit Computational Linguistics Symposium*. Springer, pp. 1–5.

Kakwani, Divyanshu et al. (2020). "IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4948–4961.

Kiparsky, Paul (2009). "On the architecture of Pāṇini's grammar". In: *Sanskrit Computational Linguistics: First and Second International Symposia Rocquencourt, France, October 29-31, 2007 Providence, RI, USA, May 15-17, 2008 Revised Selected and Invited Papers*. Springer, pp. 33–94.

Konovalov, VP and ZB Tumunbayarova (2018). "Learning word embeddings for low resource languages: the case of Buryat". In: *Komp'juternaja Lingvistika i Intellektual'nye Tehnologii*, pp. 331–341.

Krishna, Amrith et al. (2020). "Keep it surprisingly simple: A simple first order graph based parsing model for joint morphosyntactic parsing in Sanskrit". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4791–4797.

Kulkarni, Amba and KV Ramakrishnamacharyulu (2013). "Parsing Sanskrit texts: Some relation specific issues". In: *Proceedings of the 5th international sanskrit computational linguistics symposium. DK Printworld (P) Ltd.*

Kulkarni, Amba and Devanand Shukl (2009). "Sanskrit morphological analyser: Some issues". In: *Indian Linguistics* 70.1-4, pp. 169–177.

Kumar, Saurav et al. (2020). ""A Passage to India": Pre-trained Word Embeddings for Indian Languages". In: *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)*, pp. 352–357.

Levy, Omer and Yoav Goldberg (2014). "Neural word embedding as implicit matrix factorization". In: *Advances in neural information processing systems* 27.

Levy, Omer, Yoav Goldberg, and Ido Dagan (2015). "Improving distributional similarity with lessons learned from word embeddings". In: *Transactions of the association for computational linguistics* 3, pp. 211–225.

Li, Hao et al. (2018). "Visualizing the loss landscape of neural nets". In: *Advances in neural information processing systems* 31.

Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Peters, Matthew E. et al. (2018). *Deep contextualized word representations*. arXiv: 1802.05365 [cs.CL].

Robinson, Peter (2003). "The cognitive hypothesis, task design, and adult task-based language learning". In.

Sandhan, Jivnesh (2023). "Linguistically-Informed Neural Architectures for Lexical, Syntactic and Semantic Tasks in Sanskrit". In: *arXiv preprint arXiv:2308.08807*.

Schomacker, Thorben and Marina Tropmann-Frick (2021). "Language representation models: An overview". In: *Entropy* 23.11, p. 1422.

Shukla, Preeti et al. (2016). "Revival of ancient sanskrit teaching methods using computational platforms". In: *Bridging the gap between Sanskrit Computational Linguistics tools and management of Sanskrit Digital Libraries Workshop. ICON*.

Sudharshana, NP and Lina Mukhopadhyay (2022). *Task-based language teaching and assessment: contemporary reflections from across the world*. Springer Nature.

Taylor, Wilson L (1953). ""Cloze procedure": A new tool for measuring readability". In: *Journalism quarterly* 30.4, pp. 415–433.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in neural information processing systems* 30.

Wei, Jason et al. (2022). "Emergent abilities of large language models". In: *arXiv preprint arXiv:2206.07682*.

Wikipedia contributors (2023). *Pattern recognition (psychology)*. URL: https://en.wikipedia.org/w/index.php?title=Pattern_recognition_(psychology)&oldid=1175460335.

Wujastyk, Dominik (1981). “Notes On Traditional Sanskrit Teaching”. In: *South Asia Research* 1.1, pp. 30–36.