Towards Building a Computational Sense Inventory from the Monier-Williams Dictionary Using Clustering Techniques

Anagha Pradeep[1], Sriram Krishnan[1,2] and Radhika Mamidi[1]
[1] IIIT, Hyderabad
[3] Central Sanskrit University, Prayagraj
anagha.pradeep@iiit.ac.in, sriramk8@gmail.com
radhika.mamidi@iiit.ac.in

Abstract

Semantic banks, WordNets, and other sense inventories play a crucial role in the training and evaluation of many NLP applications, including machine translation and semantic understanding tasks. However, for low-resource languages such as Sanskrit, comprehensive and well-structured sense inventories are often unavailable. This not only poses a significant challenge for traditional NLP pipelines but also for leveraging state-of-the-art large language models (LLMs), which still require a foundational, machine usable sense inventory to support tasks involving meaning disambiguation and semantic consistency. Our work is motivated by this gap. We focus on creating a machine usable sense inventory for Sanskrit by using the Monier-Williams dictionary as a primary lexical resource. Our goal in building such an inventory, is to minimize ambiguity to the maximum possible extent so that the resulting senses are clear and suitable for computational models. To achieve this, we make use of clustering algorithms to group semantically similar senses into synsets and to reorganize the existing dictionary structure in a way that it better reflects semantic relationships such as polysemy and synonymy.[1]

1 Introduction

The development, training, and evaluation of Natural Language Processing (NLP) applications such as Information Retrieval (IR), Word Sense Disambiguation (WSD), and Machine Translation (MT) critically depend on the availability of sense annotated corpora such as semantic banks and WordNet. For Sanskrit, however, the availability of such resources remains severely limited (Pradeep and Mamidi, 2025). At present, the Digital Corpus of Sanskrit (DCS) (Hellwig, 2010 2025)[2] constitutes the only publicly known sense annotated corpus and even this resource is incomplete and covers only a subset of lexical items. While recent advances in large language models (LLMs) offer a pathway for automating or semi-automating sense annotation, these approaches themselves presuppose the existence of a foundational sense bank.

Our initial effort was to understand whether any of the existing lexical resources can be used as a foundational sense bank in LLMs. In particular, we examine the Monier-Williams (MW) Sanskrit–English Dictionary, a comprehensive lexical resource containing approximately 1,86,000 headwords. Rather than relying on the original XML representation of the dictionary, we make use of an aligned and structured version developed by Patel and Kulkarni (2024).

In the course of this examination, we encountered several challenges. A primary difficulty arises from dictionary entries that contain multiple senses or nested sub-entries, particularly in the case of verbs and indeclinables. This raises fundamental questions related to data modeling and sense representation: Should multi-sense entries be split into separate records, or should they be retained as unified entries? If they are divided, what principles or criteria should guide this segmentation? Conversely, if they remain unified, how can individual senses be effectively identified, accessed, and referenced within a single entry?

[1]Anagha Pradeep and Sriram Krishnan contribute equally to the paper.

[2]`http://www.sanskrit-linguistics.org/dcs/`

To better understand the internal structure of the dictionary, we examined its preface. The author states that entries are organized based on both the grammatical nature of the word and its semantic distinctions. Headwords are arranged alphabetically and may include homonyms (lexical items that share the same orthographic form but differ in derivation and meaning). For lemmas exhibiting multiple senses, the dictionary distinguishes between degrees of semantic relatedness: polysemous senses that are semantically distinct are separated using semicolons, whereas closely related senses are separated using commas. In the aligned version of the dictionary used in this study, polysemous senses are represented as separate entries, whereas closely related senses are retained within a single entry. However, we observed that this structuring principle was not applied consistently across all categories of lemmas. In particular, the treatment of polysemous and closely related senses varies across different parts of speech (noun, verb, and indeclinables), leading to irregularities in sense separation and representation. Additionally, many entries included grammatical conditions such as tense, aspect, mood and so on, that functioned as sub-keys within a lemma contributing to varied senses as well.

Keeping these issues in mind, we conduct a systematic analysis of multiple dictionary entries to identify similarities, overlaps, and divergences among the senses represented across different entries. This analysis is carried out using a widely adopted similarity metric, cosine similarity. The cosine scores are generated based on the sense embeddings obtained from 5 different encoder models. We employ 4 distinct clustering algorithms to generate clusters and analyse the performance of the models and algorithms based on a few well-known strategies. Based on the insights obtained, we attempt to reorganize and normalize the dictionary structure so that it can be directly utilized for downstream NLP tasks. The broader goal of this effort is to transform a human-oriented lexical resource into a machine-actionable sense inventory that supports computational processing.

2 Related Work

Digital versions of the lexicons are generally made use of to create WordNet, and other semantic resources such as VerbNet, PropBank, AMR banks and so on. The first WordNet was an English WordNet constructed by a group of linguists and lexicographers at Princeton, headed by George A. Miller. It made use of multiple resources such as Merriam-Webster and Oxford English Dictionary (OED) for definitions and word forms, and the Synonym Finder and Roget's International Thesaurus for semantic grouping of words (Miller et al., 1990). Unlike traditional dictionaries, a WordNet organizes words based on meaning and semantic relatedness rather than alphabetical order.

Following the Princeton WordNet, WordNets were developed for several other languages, including Finnish, Russian, Chinese, German, and Korean (Bond and Paik, 2012). In 2006, the Hindi WordNet was released as a freely available resource. Building on this foundation, the IndoWordNet project initiated at IIT Bombay expanded the framework to include multiple Indian languages, including Sanskrit (Bhattacharyya, 2010). However, the Sanskrit WordNet remains incomplete at present which limits its direct applicability to downstream NLP tasks.

2.1 The Sanskrit Sembank (SSB)

Hellwig and Biagetti (2025) presented a work on word sense annotation (WSA) for Sanskrit, resulting in the construction of a semantic bank that builds upon the Princeton WordNet for English. The annotation was carried out on the Digital Corpus of Sanskrit (DCS). DCS hosts approximately 7,00,000 sentences with lexical and morphological annotations, while dependency relations and word senses are also recorded for a subset of the corpus. DCS also hosts a revised version (a subset) of the MW Sanskrit–English Dictionary. This word sense annotation task involves linking the lemmata of the DCS dictionary with the WordNet synsets, and contextual assignment of distinct meanings to the occurrences of the lemmata in the corpus.

2.2 Other efforts

Dictionary-Based and Crowdsourced Clustering Early efforts by Kulkarni and Pedersen (2007) utilized spectral clustering and k-means on dictionary definitions, emphasizing lexically motivated features to capture semantic nuances. The challenge of establishing human baselines was addressed by Snow et al. (2010), who leveraged Amazon Mechanical Turk to decompose the task into identifying the number of general meanings and performing pairwise definition matching. Their work highlighted the importance of inter-annotator agreement and established standard evaluation metrics such as V-measure and consensus clustering for sense inventory design.

Neural Sense Representations and Induction With the advent of deep learning, representations shifted toward distributed vectors. Chen et al. (2015) proposed initializing word sense embeddings by processing WordNet glosses through Convolutional Neural Networks (CNNs), followed by context clustering. More recently, Zhang et al. (2023) utilized a Bi-LSTM architecture with self-attention to extract deep contextual features, using an optimized Density Peaks Clustering (DPC) algorithm based on cosine similarity for word sense induction. Similarly, fine-grained sense inventories have been developed through prompt-based semantic matching between English dictionaries and WordNet (Kikuchi et al., 2024).

Graph-Based and Preprocessing Methods Clustering has also been effectively used as a preprocessing step for downstream tasks. Shao et al. (2022) explored using Hierarchical Agglomerative Clustering (HAC) and Affinity Propagation (AP) alongside a Similarity-based Definition Clustering (SDC) algorithm, (closely related to the methodology in this study) to refine sense labels before performing Word Sense Disambiguation (WSD). These graph-centric approaches often rely on finding natural groupings within the lexical space to reduce sense granularity.

Evaluation, Clusterability, and Concepts Beyond algorithms, the "clusterability" of a lemma is a critical area of study. McCarthy et al. (2016) operationalized this by measuring how easily occurrences of a lemma can be partitioned into congruent clusters across different data point "views", using metrics like the variance ratio and worst-pair ratio. Recent work by Giulianelli et al. (2024) pushed towards Concept Induction, using a bi-level methodology that leverages both local lemma-centric views and global cross-lexicon perspectives with Contextualized Language Models (BERT Large). This builds upon the foundational work of Purandare and Pedersen (2004) with SenseClusters, which pioneered unsupervised discrimination by clustering the contexts of target words.

Lexicon Alignment and creation Besides these efforts, with respect to lexicon alignment, WordNet alignment, or even sub-lexicon creation, a number of works have used clustering algorithms and techniques. Charnyote et al. (2009) proposed a mechanism to build a conceptual lexicon by defining lexical models and identifying lexical concepts from an alphabetical lexicon. They discuss lexical acquisition and word-sense clustering to discover concepts and their relationships, producing candidate concepts for conceptual lexicon construction. Navigli (2006) present a wide-coverage method for clustering WordNet senses by mapping them to the coarser sense inventory of the OED, showing that automated sense clustering can achieve high accuracy when compared with manually curated gold standards.

3 Aligned Version of Monier-Williams

Among the available digital Sanskrit–English lexicons such as those by MW, Apte, Yates, and Wilson, we chose the MW dictionary for two main reasons. First, an aligned version of the MW and Apte dictionaries has been developed by Patel and Kulkarni (2024), which facilitates comparative analysis. Second, the MW dictionary provides more comprehensive lexical coverage than Apte's (see Table 1).

Table 1: Comparison of lexical coverage between Monier-Williams and Apte dictionaries.

Statistic	**MW**	**Apte**
Total number of entries	2,87,604	1,33,373
Total unique lemmas	1,94,072	31,085
Unique lemmas exclusive to the dictionary	1,78,011	15,024

The MW dictionary is alphabetically organized according to the Devanāgarī script and follows a four-line structure: the primary Nāgarī line lists roots and key Sanskrit words; the secondary Indo-Romanic line shows derivative and cognate word series; the third line groups compounds formed from a leading word; and the fourth Indo-Italic line lists compounds formed from a leading compound. In the aligned version used in this work, each of these lines is represented as a distinct entry, with derivational or segmentation information provided in a separate column.

With respect to sense representations, the MW distinguishes identical word forms with different meanings by numbered transliterations. For lemmas with multiple senses, minor meaning variations are separated by commas, while clearly distinct senses are separated by semicolons, except within parentheses. This structure is partially modified in the aligned version. The aligned version stores identical word forms with different meanings as separate entries just like the original, and retains meanings having minor variations as separated by commas, while clearly distinct senses are stored as separate entries for as many as possible, mostly covering the stems (prātipadikas). But, for roots (dhātus) and indeclinables (avyayas), we observed that such a splitting was not done.

The DCS corpus also includes a dictionary whose meanings are derived from a digitized version of the MW dictionary.[3] The structural difference between the aligned version of the dictionary we are using, and the DCS dictionary is that, the DCS dictionary combines both minor variations and clearly distinct senses within a single entry, separating them uniformly by semicolons. For instance for the lemma "aṃśa", the aligned version has 10 different entries, and the DCS dictionary has one entry consisting of all the senses.[4] We inferred this merging might have been a reason for the reduction in the total number of entries between the aligned version and the DCS dictionary, with DCS having 1,79,806 entries. However, the number of unique lemmas in DCS was 1,64,202, which accounts to almost 30,000 lemmas missed in DCS when compared to the aligned version. This discrepancy further reinforced our decision to use the aligned version for our work.

4 Data

As mentioned earlier, we used the aligned version of the MW dictionary for our task. However, the raw data contained a substantial amount of noise, making an extensive cleaning process necessary. This noise included cross-referential expressions such as quod vide (which see), confer (compare), see above, see below, and similar annotations, all of which had to be carefully removed. Additionally, the entries contained grammatical information related to class, tense, aspect, mood, and case. Since this information was not intended to be part of the senses, we excluded it from the list of senses and retained it in a separate column for potential future use. We further introduced a semi-automatic categorical annotation for each lemma to obtain a high-level understanding of the dataset, specifically the homonyms, and to facilitate category specific cleaning. The lemmas were classified into five categories: **prātipadika** (stems), **dhātu** (roots), **nāma-dhātu** (nominal verbs), **cpd-prātipadika** (compounds), **upasarga** (preverbs), **upasarga-dhātu** (roots prefixed with preverbs) and **avyayas** (indeclinables). A list of all **dhātus** (roots) was collected from the Sanskrit Heritage Platform (SH)[5] and compared with

[3]`https://www.sanskrit-lexicon.uni-koeln.de/scans/MWScan/2020/web/index.php.`

[4]share; portion; part; party; partition; inheritance; share of booty; earnest money; stake (in betting); denominator of a fraction; degree of latitude or longitude; day; Name of an Aditya.

[5]`https://sanskrit.inria.fr/`

the MW lemmas for marking the dhātu category. MW dictionary entries record the class and pada (parasmaipada vs ātmanepada) information which were mainly used for matching the MW lemmas with the SH roots. The category **upasarga-dhātu** was marked for all dhātu entries whose derivation had the preverb attached to the dhātu. MW marks all nāma-dhātus as 'Nominal Verb' in their meanings. **upasargas** and **avyayas** were marked based on a collected list of preverbs and indeclinables. The remaining entries were marked as **prātipadika**. **cpd-prātipadika** was marked wherever the prātipadika is split by a hyphen.

The original aligned version consisted of 2,87,604 entries, which was reduced to approximately 2,64,700 entries after cleaning. Despite the time consuming nature of this process, we estimate that the cleaned dataset may still contain around 10% noise. For our experiments and analysis, we therefore worked with a carefully selected subset rather than the entire dataset. This subset was constructed by selecting frequently occurring lemmas with a high number of senses from the DCS corpus, along with a few additional lemmas observed to exhibit high polysemy. The final subset comprised of 54 lemmas that accounted for 568 entries. The lemmas were chosen across different categories and consisted of: 16 dhātus, 4 avyayas, 2 nāma-dhātus, 2 cpd-prātipadikas, and 30 prātipadikas. All experiments, analyses, and discussions in this work are based on this subset. Nevertheless, we extend our experiments to the full dataset and release it as part of this work.

5 Experiments

The primary dataset contains the dictionary meanings for each <head, lemma, derivation, category>. The meanings corresponding to the same <head, lemma, derivation, category> are considered for clustering, as they represent the sample space of synonymous and polysemous senses for that particular lemma with the specified derivation and category. This ensures that homonyms, that happen to share a lemma but belong to different morphological or grammatical derivations, are filtered out in the initial step. The details of the experiments conducted to evaluate the clustering of dictionary meanings through various embedding models and graph-based clustering algorithms are discussed ahead. Figure 1 depicts the pipeline of the sense clustering experiments.

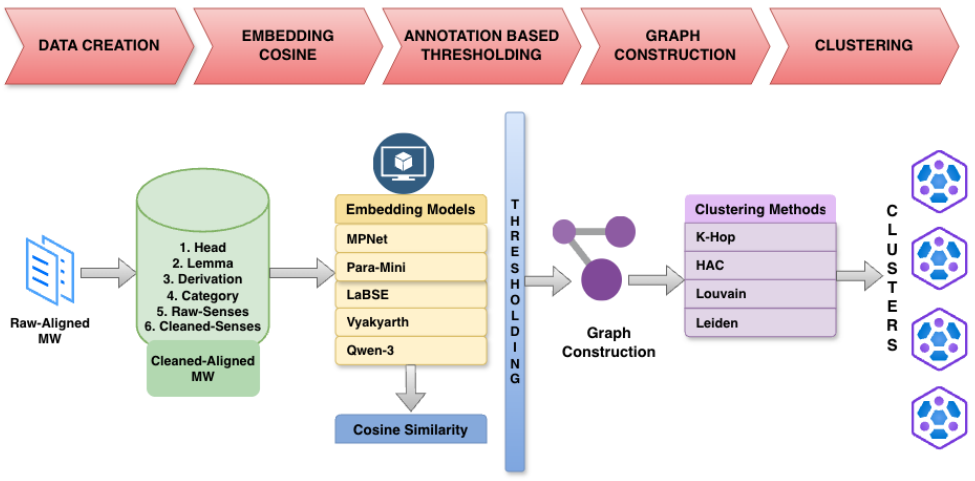

Figure 1: Pipeline of the Sense Clustering experiments

5.1 Embeddings:

We utilize five distinct transformer-based encoder models to generate sense embeddings, each offering a different architectural bias or training objective:

- **MPNet** (Song et al., 2020): This model utilizes a unified architecture that integrates the benefits of *Masked Language Modeling* (MLM) and *Permuted Language Modeling* (PLM).

By leveraging the full position information of a sentence, it captures complex dependencies while avoiding the pretrain-finetune discrepancy found in earlier models.

- **LaBSE** (Feng et al., 2022): The *Language-Agnostic BERT Sentence Embedding* model is pre-trained on 109 languages using a dual-encoder architecture. It effectively maps multilingual text into a shared semantic space, which is critical for our case as the dictionary meanings have both English and Sanskrit words.

- **Paraphrase-MiniLM** (Reimers and Gurevych, 2019): A distilled, lightweight version of large-scale transformers, this model is trained specifically on vast paraphrase datasets.

- **Qwen** (Qwen, 2025): As a modern LLM variant, Qwen leverages extensive pre-training on diverse datasets to capture nuanced contextual relationships. Its latent representations offer a deep semantic understanding that accounts for complex linguistic structures often missed by smaller encoders.

- **Vyakyarth** (Singh et al., 2024): Vyakyarth is an advanced sentence embedding model designed for Indic languages (10+ including Sanskrit), built upon the STSB-XLM-R-Multilingual architecture. Fine-tuned using contrastive learning, Vyakyarth excels in cross-lingual sentence similarity, semantic search, multilingual retrieval, and text clustering.

5.2 Clustering Mechanism

Thresholding via Manual Annotation In order to determine the optimal similarity threshold, a manual annotation exercise was conducted by two annotators[6], on a subset of five extremely polysemous lemmas (comprising of 80 entries). For every annotated cluster, the minimum cosine similarity scores assigned by a model to all pairs within that cluster were calculated. The median of these minimum scores across the sample was calculated for each annotator and averaged. This model-specific thresholding ensures that we do not depend on a single hard threshold for all the models as different models exhibit varying ranges of cosine similarities.

Annotation agreement To ensure the agreement levels between annotators, we computed an exact match agreement where both annotators' cluster annotations are parsed into unordered sets of clusters, and agreement is counted when the two parsed clusterings are exactly identical for an item. The inter-annotator agreement score is computed as the average of this binary agreement across all items. For a sample set consisting of five lemmas and 307 senses, the annotators obtained a score of 0.81.

Clustering Algorithms Based on the annotation clusters, we observed that the data points are not too dense, thus ruling out density-based clustering algorithms. We incorporated a revised version of the **K-hop Clustering**. This is a graph-based approach where nodes are grouped based on proximity within k edges from a seed node. Unlike standard overlapping neighborhood methods, our implementation follows a *greedy disjoint* strategy i.e., once a sense is assigned to a cluster through a k-hop path, it is excluded from all subsequent clusters, ensuring a unique partition for every sense in the lemma group. Additionally, we tested the following clustering algorithms:

- **Hierarchical Agglomerative Clustering (HAC):** Agglomerative clustering is a hierarchical method in unsupervised machine learning that groups similar data points into clusters. It follows a bottom-up process, beginning with each data point as an individual cluster and progressively merging the most similar clusters according to a chosen distance metric and linkage criterion (Abdine et al., 2023). In this study, we employ *average linkage*, i.e., merging clusters by averaging the distance between the points, to maintain a balance between cluster cohesion and separation.

[6]Both linguists, with a sound knowledge in Sanskrit.

- **Louvain Community Detection:** A multi-level modularity optimization algorithm that assigns nodes to communities to maximize the density of internal edges compared to a random graph model (Traag et al., 2019). It is particularly efficient for large-scale semantic networks where the number of clusters is not known priorly (Waltman and Van Eck, 2013).
- **Leiden Algorithm:** An advanced iteration of the Louvain method designed to address the resolution limit and ensure that all identified communities are internally well-connected. It has been specifically designed to address the problem of badly connected communities. The Leiden algorithm provides more stable and cohesive partitions by guaranteeing that clusters cannot be further subdivided into significantly better sub-communities (Traag et al., 2019).

All possible combinations of the five embedding models and four clustering algorithms were executed independently. The similarity matrix (graph) for each <head, lemma, derivation, category> group was constructed by calculating pairwise cosine similarities. Edges were marked only when the similarity exceeded the model-specific threshold derived earlier. This resulted in 20 distinct model-algorithm configurations, providing a comprehensive matrix for further analysis.

6 Evaluation

The 20 distinct model-algorithm configurations behaved variedly across models and algorithms. Figure 2 illustrates this variation by depicting the average number of clusters formed per lemma. To gain a clearer understanding of how the different models and algorithms performed in forming clusters, we applied two evaluation strategies. The first strategy employs four distinct metrics that focus exclusively on the clustering methods and their behavior. While the second strategy evaluates the generated clusters by comparing them with the gold-standard annotations across the entire dataset of 54 lemmas. These strategies are discussed below.

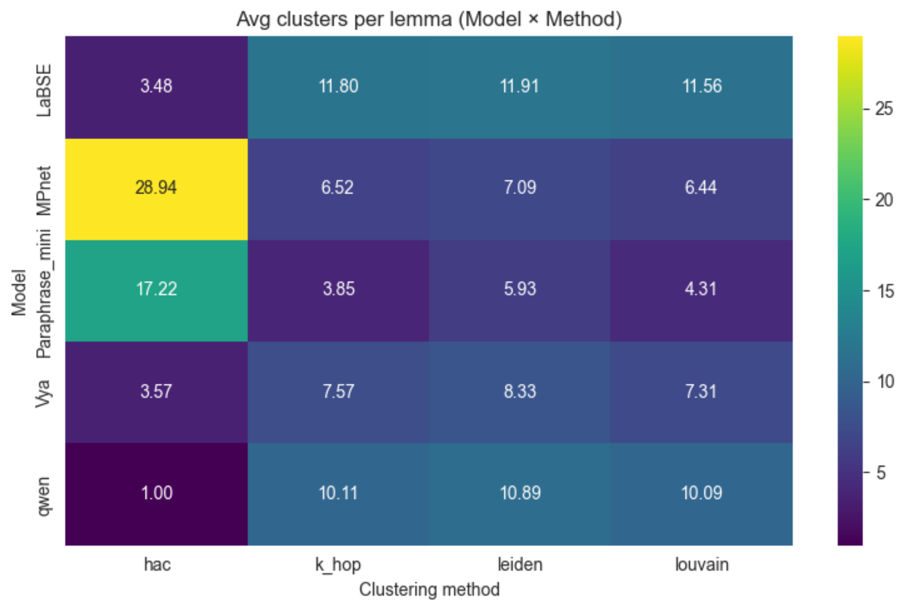

Figure 2: Heatmap depicting the average number of clusters per lemma heatmap

6.1 Intrinsic Evaluation

We choose 4 different cluster assessments: pairwise-dominance, Shannon entropy, transitivity score and anchor separation score. While the first two assessments focus on the clustering methods themselves, the latter two focus on the clustering behaviour.

1. **Pairwise-Dominance:** Through this strategy, we try to understand which of the clustering algorithms performed better than the other. We compare each pair of clustering algorithms

on three statistics: standard deviation, median, and degeneracy, and award points for better performance. Based on this we decide a win, loss or tie. We then count the total wins across all pairwise comparisons to produce an overall ranking and the best algorithm. In our comparisons, the Louvain algorithm performed the best, winning against K-hop and HAC (see Table 2).

Table 2: Pairwise Dominance Summary of Clustering Algorithms

Algorithm	**Wins**	**Losses**	**Ties**	**Beats**
K-hop	1	1	1	HAC
HAC	0	3	0	–
Louvain	**2**	0	1	K-hop, HAC
Leiden	1	0	2	HAC

2. **Shannon Entropy:** Entropy scores help us to see how stable and balanced a clustering algorithm is. For each clustering algorithm, the scores are grouped into a histogram, converted into a probability distribution, and evaluated using entropy to quantify how dispersed and diverse the scores are. Our results (given in Table 3) show that three of the four algorithms (K-hop, Louvain, and Leiden) exhibit identical entropy scores, while HAC displays a comparatively lower entropy score.

Table 3: Entropy-Based Ranking of Clustering Algorithm

Algorithm	**Entropy Score**
K-hop	**1.609**
Louvain	**1.609**
Leiden	**1.609**
HAC	0.950

3. **Global Graph Transitivity:** Transitivity measures the triangle density of a graph i.e., how often pairs of nodes that are connected to a common node are also connected to each other. To compute this, we first construct a graph for each item, where senses are represented as nodes, and edges are created when the cosine similarity meets or exceeds the model-specific threshold. For each clustering algorithm, we then evaluate the transitivity of each cluster's graph. We report the mean and median per model-algorithm pair and then average the medians across the algorithms. The model that has the highest median then is inferred to be handling transitivity very well. Among our selected models, the Paraphrase-Mini, has the highest median average, as given in Figure 3. We also report the mean and median scores for all model-algorithm pairs in Table 4.

Table 4: Transitivity Scores across Embedding Models and Clustering Algorithms: Best algorithm (given a model) is in (**Bold**), and Best model (given an algorithm) is (Underscored)

Model	**K-hop**		**HAC**		**Louvain**		**Leiden**	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
MPNet	0.493	0.501	1.000	**1.000**	0.648	0.651	0.661	0.637
LaBSE	0.508	0.493	0.446	0.508	0.618	**0.625**	0.620	**0.625**
Para-Mini	0.650	0.675	0.998	**1.000**	0.757	0.749	0.798	0.809
Qwen	0.524	0.556	0.533	0.557	0.608	0.635	0.627	**0.636**
Vya	0.592	0.588	0.518	0.561	0.701	0.688	0.750	**0.739**

4. **Anchor Separation:** Anchor separation scores tell us how well each cluster is separated from the rest. It predicts the strength of each cluster and about how tightly it is formed. To

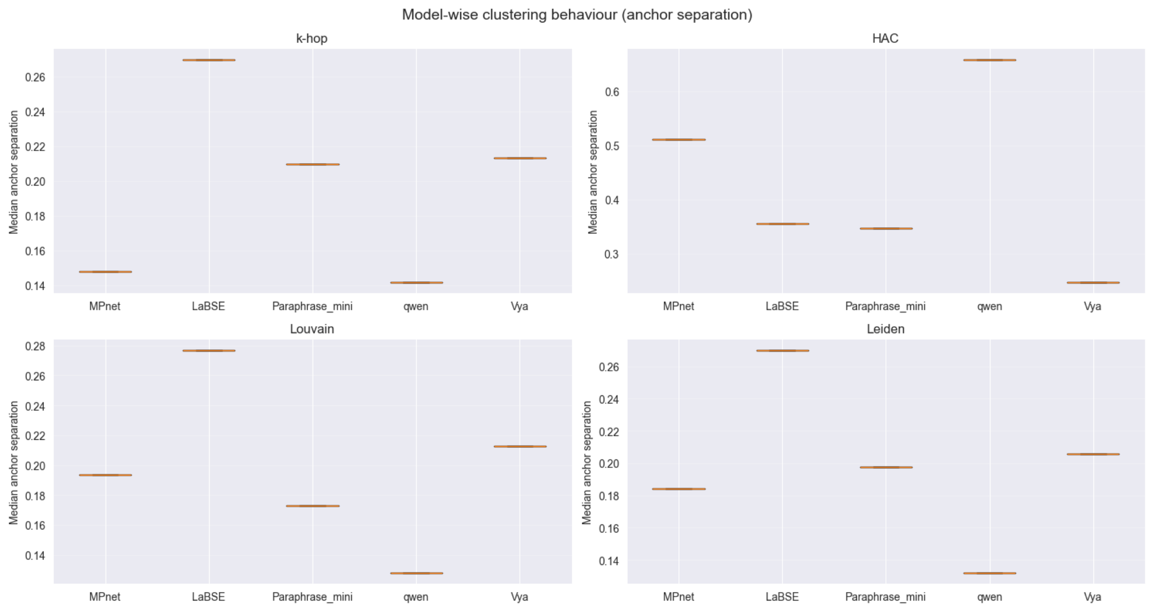

Figure 3: Average Transitivity Median

estimate this, for each cluster, we compute: Intra-cluster similarity–how similar the nodes inside the cluster are to each other, and Inter-cluster similarity–how similar those nodes are to nodes outside the cluster. Anchor separation score is calculated as the difference between the inter and intra-cluster similarity. Similar to transitivity ranking, we average the anchor separation score across clusters, aggregate scores per clustering algorithm and model, and report the mean and median separation, plus an overall summary. The model that ranks top for anchor separation is LaBSE with a median of 0.29. A pairwise look-up of the model-algorithm is given in Table 5.

Table 5: Anchor Separation Scores across Embedding Models and Clustering Algorithms: Best algorithm (given a model) is in (**Bold**), and Best model (given an algorithm) is (<u>Underscored</u>)

Model	**K-hop**		**HAC**		**Louvain**		**Leiden**	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
MPNet	0.168	0.148	0.496	**0.510**	0.204	0.193	0.204	0.184
LaBSE	0.300	<u>0.270</u>	0.388	**0.355**	0.310	<u>0.277</u>	0.303	<u>0.270</u>
Para-Mini	0.253	0.210	0.350	**0.347**	0.210	0.173	0.220	0.198
Qwen	0.181	0.142	0.655	<u>**0.658**</u>	0.176	0.128	0.174	0.132
Vya	0.252	0.213	0.312	**0.247**	0.239	0.213	0.235	0.206

Based on all 4 strategies mentioned above, we calculate an Intrinsic Score to identify the top 5 model-algorithm pairs. For each pair, scores from the four metrics are normalized and an intrinsic score is calculated using predefined weights:

$$\text{IS}_i = 0.3 \cdot P_{\text{norm},i} + 0.2 \cdot H_{\text{norm},i} + 0.3 \cdot T_{\text{norm},i} + 0.2 \cdot S_{\text{norm},i} \tag{1}$$

where for the i-th model-algorithm combination:

- IS_i is the calculated Intrinsic Score.
- $P_{\text{norm},i}$ is its normalized Pairwise Dominance.
- $H_{\text{norm},i}$ is its normalized Shannon Entropy.
- $T_{\text{norm},i}$ is its normalized Transitivity.

- $S_{\text{norm},i}$ is its normalized Anchor Separation.

Finally we rank the algorithms by the intrinsic score to get an overall best-performing configuration. For our sample data, the Paraphrase-Mini (model) and Louvain (algorithm) combination achieves the highest performance with an intrinsic score of 0.66. The scores for the top five pairs are given in Table 6.

Table 6: Top-ranked model–clustering combinations based on Intrinsic Score.

Model	**Algorithm**	**Intrinsic Score**
Para-mini	Louvain	**0.669**
Vya	Louvain	0.647
LaBSE	Louvain	0.634
MPnet	Louvain	0.618
Qwen	Louvain	0.584

6.2 Extrinsic Evaluation

We compare the clusters produced by each model–algorithm combination with the human-annotated clusters using three metrics: B-cubed F-Score ($F\text{-}B^3$), Adjusted Rand Index (ARI), and Adjusted Mutual Information (AMI).

The $F\text{-}B^3$ metric measures clustering performance element-wise by computing precision and recall for each data point, assessing whether it has been grouped with its correct semantic neighbors or not. The ARI evaluates clustering quality at the pairwise level by quantifying how consistently pairs of items are either placed in the same cluster or separated into different clusters (true positives and true negatives), with adjustment for agreement expected by chance. The AMI assesses clustering similarity at the distributional level by using Shannon entropy to measure the shared information between predicted clusters and ground truth labels, and is also corrected for chance agreement. For each model-algorithm pair, the raw scores of the three metrics were Min-Max normalized and averaged to compute the overall Extrinsic Score (ES), which is subsequently used to rank the combinations, as reported in Table 7.

Table 7: Extrinsic Cluster Assessment against Gold Annotation Ranking

Model_Algorithm	$F\text{-}B^3$	**ARI**	**AMI**	**Extrinsic Score**
LaBSE_leiden	0.2904	0.3675	0.6667	**1.000**
LaBSE_louvain	0.2766	0.3588	0.6541	0.965
Para-Mini_hac	0.2401	0.2979	0.7138	0.893
LaBSE_k_hop	0.2297	0.2969	0.6257	0.820
Qwen_leiden	0.2308	0.2880	0.6210	0.809

6.3 Selecting the Best Model-Algorithm Combination

It is evident that there is a difference in the intrinsic and extrinsic ranking results, which makes it hard to choose one best model-algorithm pair. Therefore, we decided to apply min-max normalization to each model-algorithm pair mapping both intrinsic and extrinsic results. We compute a weighted composite score (CS) giving a higher weightage to the extrinsic score, mainly for the final sense inventory to align well with the gold annotations. The macro-averaged composite score is used to determine the best model-algorithm pair, which is reported in table 8.

7 Analysis & Discussion

The quality of the sense inventory thus created is determined by several factors, with every stage of the pipeline significantly contributing to it. We discuss here the inferences from the results and

Table 8: Min-Max Normalization Scores mapping Extrinsic and Intrinsic Evaluation results

Model	Algorithm	Extrinsic	Intrinsic	Composite Score
LaBSE	louvain	0.965	0.941	**0.955**
LaBSE	leiden	1.000	0.680	0.872
qwen	louvain	0.766	0.855	0.802
Vya	louvain	0.679	0.963	0.792
MPnet	louvain	0.695	0.914	0.783

analysis, limitations with respect to data, models, algorithms, and evaluation metrics, possible challenges to extend this to the entire MW dictionary or any other Sanskrit-English dictionary, along with qualitative comparisons with the original versions.

7.1 Data

The major challenge for such a sense inventory creation lies in the nature of the data, in our case, the MW data. The dictionary is 'noisy' from a computational perspective with respect to sense-based tasks. It contains a dense mix of definitions, citations, and grammatical markers, with the senses having a high degree of variance in the sense length. On the other hand, such information is useful for disambiguation tasks.

Extracting the senses alone from a list of meanings with too much information is extremely strenuous as a manual task. While LLMs are the immediate solutions we turn to, for solving such problems, we formulated this task for an effective evaluation of LLM's results. Additionally, measuring the error rate remains difficult and making use of automated cleaning is highly risky.

The Sanskrit-English Ratio: The dual nature of our data–Sanskrit words within English definitions–does create a cross-lingual dependency. This specific setup of a hybrid dictionary meanings calls for the Sanskrit-understanding of the models to be sensitive to cross-lingual data. In our case, not all models are sensitive to cross-lingual data.

Categories The classification of the lemmas based on Sanskrit grammatical categories is relatively a coarse-grained effort which majorly focused on the roots (dhātu) and indeclinables (avyaya) as they had extremely high density of number of senses per entry. While some categories were annotated based on fixed lists of lemmas, some were marked using carefully curated pattern matching techniques, especially for the nominal verbs (nāmadhātu). As all the remaining entries were annotated as prātipadika, this could be erroneous and difficult to detect.

7.2 Model Performance

One major question with respect to models is: can we unambiguously say that a particular model captures the senses perfectly? We are majorly using sentence-based transformers to capture senses which could be a word, a phrase, or a sentence. We did experiment with various other models like MiniLM-L6, MiniLM-L12, RoBERTa-Large, IndicBERT and Sarvam. Sarvam, being a decoder-only model, performed poorly in our training-free sense clustering setup, which led us to focus on encoder-based architectures. Although IndicBERT is an encoder-only model, its performance was also unsatisfactory and was therefore excluded from our experiments.

Among MiniLM-L6, MiniLM-L12, MPNet, and RoBERTa-Large, MPNet demonstrated better performance compared to the others, and we selected it as our baseline model.

Paraphrase-Mini and Vyākyārth Despite being a distilled model, the predominance of Paraphrase-MiniLM is due to its training on paraphrase tasks making it exceptionally sensitive to 'meaning-overlap'. On the other hand, Vyākyārth's better performance is likely due to its exposure to Sanskrit, allowing to capture the senses with Sanskrit words.

Thresholding A major highlight of this methodology is the median-based thresholding. Although we curated the model-specific thresholding for the algorithms, these are derived from a small annotated set. While they proved effective here, a larger gold standard would be required to confirm if these thresholds generalize across all 260k+ entries. Alternatively, a multi-level scaling strategy should be deployed, where the annotated data size is increased incrementally, and the variation of the threshold can be observed to determine whether a fixed universal threshold is viable or an adaptive strategy is required as we increase the data size.

Capturing Discrete Senses Our goal is to capture similar senses of the same lemma together and keep apart distinct senses. While this methodology helped to an extent, this also introduced the problem of multiple senses of the same English definition, especially for the single word dictionary senses (for example, "being"). So, without explicit context (sentence-level data), the models rely entirely on the definition's wording.

Qualitative Analysis To qualitatively analyse the performance of the models and clustering algorithms, we focus on the lemma ***hari***, which is highly polysemous and appears in both the sample dataset and the annotated sample. We can see from Table 9 that the original dataset contains 31 clusters, while the annotation sample expands this to 48 clusters, indicating that manual annotation introduces finer sense distinctions compared to the source grouping. MPNet (46) and Para-Mini (31) with HAC closely match the annotation sample, while all other combinations drastically under-cluster, collapsing many senses into very few groups. A likely reason for this behaviour is the high cosine similarity between certain senses, such as "name of Indra", "name of Brahmā", "name of Śiva", and "name of Śukra." Although these are required to be treated as distinct entries, the embedding models appear to map them closely. This proximity appears to drive inconsistent clustering outcomes. In some instances, senses such as "Name of a son of Akampana," "Name of a son of Garuḍa," "Name of a son of Parājit," "Name of a son of Parāvṛt," and "Name of a son of Tārakākṣa" were merged into a single cluster while senses like "Name of a king of the Nāgas" and "Name of a mountain; Name of a school; Name of a song" were distributed across separate clusters. These patterns suggest that the models rely heavily on surface-level lexical similarity, rather than consistently capturing the finer-grained distinctions required by the dataset.

Similarly, senses of ***hari*** denoting different animal kinds (e.g., frog, goose, horse, jackal, lion, monkey) are also likely to be embedded in overlapping regions. As a result, the clustering algorithms struggle to separate these senses effectively. This highlights potential limitations both in the quality of the generated embeddings and in the reliance on cosine similarity as the primary metric for clustering.

Table 9: Cluster counts for lemma *hari* (ha/ri, prātipadika)

Model / Source	**Clusters**	**K-hop**	**HAC**	**Louvain**	**Leiden**
Original	31	–	–	–	–
Annotation Sample	48	–	–	–	–
LaBSE	–	22	2	20	20
MPNet	–	10	46	7	6
Para-Mini	–	10	31	7	7
Qwen	–	16	1	15	15
Vya	–	15	7	16	16

7.3 Algorithmic Behaviour

Clustering algorithms are aplenty and we have to choose or design our own approach based on our requirement. We experimented with other clustering algorithms like Spectral, Affinity Propagation, and Hub-based clustering, but their performances on the specific sample data, with the same models were poor. Amongst the chosen algorithms, we observe that the modularity-based Louvain and Leiden emerged as the most robust performers. On the other hand, HAC tends to create tiny isolated cliques, leading to extreme over-segmentation, thus failing to capture synonymy properly.

7.4 Evaluation Metrics

We adopted two evaluation strategies: intrinsic and extrinsic, to ensure a comprehensive assessment of our clustering approach. The intrinsic evaluation strategy focused on assessing the internal quality of the generated clusters, without reference to any external ground truth. This allowed us to compare how different model-algorithm pairs performed relative to one another, purely in terms of their clustering behavior. However, intrinsic evaluation does not determine whether the generated clusters align with the intended or correct groupings.

To address this limitation, we also applied the extrinsic evaluation strategy, which compares the generated clusters against human-annotated (gold standard) clusters. By combining both intrinsic and extrinsic evaluations, we were able to assess not only the structural quality of the clusters and the relative performance of model–algorithm pairs, but also their alignment with the annotated ground truth.

In the intrinsic evaluation, we observed that the Louvain algorithm consistently outperformed the other algorithms, regardless of the underlying model. In contrast, the extrinsic evaluation showed that all algorithms were relatively competitive.

To determine the best model–algorithm pair, we considered both intrinsic and extrinsic performance. However, while structural optimality is important, we placed greater weightage on extrinsic evaluation, since our primary objective was to select a final model–algorithm combination that aligned closely with the gold-standard annotations.

7.5 Other issues

Singleton problem Many algorithms produced "singleton" clusters. Determining whether a singleton represents a true homonym (a totally distinct meaning) or a failure to cluster a polyseme is a major evaluation hurdle. In fact, from Table 5 we can observe that the HAC algorithm has the highest median scores for all models, depicting very good anchor separation. However, we infer that this apparent advantage is largely attributable to HAC's tendency to generate an unusually high number of singleton clusters.

The Sanskrit-English problem In our manual analysis of 5 selected lemmas, we found that models occasionally struggle with "metaphorical" senses. A theoretical analysis of how the sense spectrum is according to various linguistic phenomena like metaphor is to be explored further.

8 Conclusion & Future Work

This work presents a novel attempt at clustering senses from the MW dictionary to construct a structured sense inventory suitable for computational tasks such as WSD, Machine Translation and Information Retrieval. To the best of our knowledge, this is the first systematic effort to reorganize MW dictionary senses into a machine-usable sense inventory through clustering-based methods. Beyond the methodological contribution, we release the resource at three distinct levels: (i) a base aligned version, (ii) the annotated sample set and the sample set used for experiments and analysis, and (iii) the full version containing all entries.[7]

While this work establishes a foundational sense inventory, several important extensions remain open for future work. The first and the primary efforts would be directed towards comparing the reformed sense inventory to the actual aligned dictionary. Another key direction is to focus on enriching the sense inventory with explicit grammatical information and additional semantic relations such as hypernymy, hyponymy, and meronymy, with the goal of building

[7]The resource is publicly available on GitHub at: `https://github.com/Anaghapradeep21/MW-Sense-Inventory`

a full-fledged Sanskrit-English WordNet. We plan to compare our base (aligned) version of the dictionary with other existing resources including DCS, Apte's dictionary, and Sanskrit Sembank, and to better understand how these differ in structuring homonymy, polysemy, and synonymy. We also want to try and use contextual data from DCS corpus for learning-based or context-based sense modeling. Finally, scaling the clustering approach with adaptive thresholding and exploring alternative evaluation strategies, such as contrastive evaluation, remain important directions for future work.

Acknowledgements

We would like to thank Dr. Dhaval Patel for sense aligned MW dataset. We extend our gratitude to Prof. Amba Kulkarni for her insights on various aspects of dictionary structure and lexicography. We also appreciate Ajit Srikant and Sushvin Marimuthu, for their insights on clustering algorithms and evaluation metrics.

References

Hadi Abdine, Moussa Kamal Eddine, Davide Buscaldi, and Michalis Vazirgiannis. 2023. Word sense induction with agglomerative clustering and mutual information maximization. *AI Open*, 4:193–201.

Pushpak Bhattacharyya. 2010. IndoWordNet. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May. European Language Resources Association (ELRA).

Francis Bond and Kyonghee Paik. 2012. A survey of wordnets and their licenses. In *proceedings of the 6th global WordNet conference (GWC 2012)*, pages 64–71. Matsue.

Pluempitiwiriyawej Charnyote, Cercone Nick, and An Xiangdong. 2009. Lexical acquisition and clustering of word senses to conceptual lexicon construction. *Computers & Mathematics with Applications*, 57(9):1537–1546.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2015. Joint learning of character and word embeddings. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*. Relevant for gloss composition and context clustering methodologies.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland, May. Association for Computational Linguistics.

Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2024. To word senses and beyond: Inducing concepts with contextualized language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, St. Julians, Malta, March. Association for Computational Linguistics.

Oliver Hellwig and Erica Biagetti. 2025. The sanskrit sembank. *Language Resources and Evaluation*, pages 1–24.

Oliver Hellwig, 2010–2025. *The Digital Corpus of Sanskrit (DCS)*.

Masato Kikuchi, Masatsugu Ono, Toshioki Soga, Tetsu Tanabe, and Tadachika Ozono. 2024. Coarse-grained sense inventories based on semantic matching between english dictionaries. In *2024 11th International Conference on Advanced Informatics: Concept, Theory and Application (ICAICTA)*, pages 1–6.

Aniruddha Kulkarni and Ted Pedersen. 2007. Leveraging dictionary definitions for word sense discrimination. In *Proceedings of the 11th Conference on Computational Natural Language Learning (CoNLL-2007)*, pages 74–81, Prague, Czech Republic, June. Association for Computational Linguistics.

Diana McCarthy, Marianna Apidianaki, and Katrin Erk. 2016. Word sense clustering and clusterability. In *Proceedings of the Fourth International Workshop on Metaphor in NLP*, San Diego, California, June. Association for Computational Linguistics.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to wordnet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

Roberto Navigli. 2006. Meaningful clustering of senses helps boost word sense disambiguation performance. In Nicoletta Calzolari, Claire Cardie, and Pierre Isabelle, editors, *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 105–112, Sydney, Australia, July. Association for Computational Linguistics.

Dhaval K Patel and Amba Kulkarni. 2024. Word sense alignment of Sanskrit lexica. In Arnab Bhattacharya, editor, *Proceedings of the 7th International Sanskrit Computational Linguistics Symposium*, pages 1–13, Auroville, Puducherry, India, February. Association for Computational Linguistics.

Anagha Pradeep and Radhika Mamidi. 2025. Sandarśana: A survey on sanskrit computational linguistics and digital infrastructure for sanskrit. *ACM Comput. Surv.*, 57(10), May.

Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in Low and High dimensional spaces. In *Proceedings of the Eleventh Conference on European Chapter of the Association for Computational Linguistics (EACL)*, Barcelona, Spain, April. Association for Computational Linguistics.

Qwen. 2025. Qwen3 technical report.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 11.

J. Shao, P. Bakx, and K. Bontcheva. 2022. Word sense disambiguation using clustered sense labels. In *Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, Marseille, France. European Language Resources Association.

Pushkar Singh, Sandeep Kumar Pandey, and Rajkiran Panuganti. 2024. Vyakyarth: A multilingual sentence embedding model for indic languages. GitHub.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2010. Clustering dictionary definitions using Amazon Mechanical Turk. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1138–1147, Cambridge, MA, October. Association for Computational Linguistics.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: masked and permuted pre-training for language understanding. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12.

Ludo Waltman and Nees Jan Van Eck. 2013. A smart local moving algorithm for large-scale modularity-based community detection. *The European physical journal B*, 86(11):471.

Yuan Zhang, Xia Wang, and Zhou Li. 2023. A method for constructing word sense embeddings based on word sense induction using Bi-LSTM and DPC. *Applied Intelligence*, 53.