A APPENDIX

In this supplementary material, we present various details omitted from the main text due to lack of space, including a proof of Thm 3.1, a detailed analysis of the time complexity of the various modules in the training and prediction pipelines of DECAF, details of the datasets and evaluation metrics used in the experiments, further clarifications about how some ablation experiments were carried out, as well as additional experimental results including a subjective comparison of the prediction quality of DECAF and various competitors on handpicked recommendation examples.

A.1 Proof of Theorem 3.1

We recall from the main text that $\mathcal{L}(\Theta)$ denotes the original likelihood expression and $\tilde{\mathcal{L}}(\Theta \mid \mathcal{S})$ denotes the approximate likelihood expression that incorporates the shortlister $\mathcal{S}$. Both expression are reproduced below for sake of clarity.

$$\mathcal{L}(\Theta) = \frac{1}{NL} \sum_{i \in [N]} \sum_{l \in [L]} \ell_{il}(\Theta)$$

$$\tilde{\mathcal{L}}(\Theta \mid \mathcal{S}) = \frac{K}{NLB} \sum_{i \in [N]} \sum_{l \in \mathcal{S}(\hat{\mathbf{x}}_i)} \ell_{il}(\Theta)$$

Theorem A.1 (Theorem 3.1 Restated). *Suppose the training data has label sparsity at rate s i.e. $\sum_{i \in [N]} \sum_{l \in [L]} \mathbb{I}\{y_{il} = +1\} = s \cdot NL$ and the shortlister offers a recall rate of r on the training set i.e. $\sum_{i \in [N]} \sum_{l \in \mathcal{S}(\hat{\mathbf{x}}_i)} \mathbb{I}\{y_{il} = +1\} = rs \cdot NL$. Then if $\hat{\Theta}$ is obtained by optimizing the approximate likelihood function $\tilde{\mathcal{L}}(\Theta \mid \mathcal{S})$, then the following always holds*

$$\mathcal{L}(\hat{\Theta}) \leq \min_{\Theta} \mathcal{L}(\Theta) + O\left(s(1-r)\ln\left(\frac{1}{s(1-r)}\right)\right).$$

Below we prove the above claimed result. For the sake of simplicity, let $\Theta^* = \arg\min_{\Theta} \mathcal{L}(\Theta)$ denote the optimal model that could have been learnt using the original likelihood expression. As discussed in Sec 3, OvA methods with linear classifiers assume a likelihood decomposition of the form $\mathbb{P}[\mathbf{y}_i \mid \mathbf{x}_i, \Theta] = \prod_{l=1}^{L} \mathbb{P}[y_{il} \mid \hat{\mathbf{x}}_i, \mathbf{w}_l] = \prod_{l=1}^{L} (1 + \exp(y_{il} \cdot \langle \hat{\mathbf{x}}_i, \mathbf{w}_l \rangle))^{-1}$ where $\hat{\mathbf{x}}_i = \text{ReLU}(\mathcal{E}(\mathbf{x}_i))$ is the document embedding obtained using token embeddings $\mathbf{E}$ and embedding block parameters taken from Θ, and $\mathbf{w}_l$ is the label classifier obtained as shown in Fig 3. Thus, for a label-document pair $(i, l) \in [N] \times [L]$, the model posits a likelihood

$$\mathbb{P}[y_{il} \mid \hat{\mathbf{x}}_i, \mathbf{w}_l] = (1 + \exp(y_{il} \cdot \langle \hat{\mathbf{x}}_i, \mathbf{w}_l \rangle))^{-1}$$

However, in the presence of a shortlister $\mathcal{S}$, the above model fails to hold since for a document i, a label $l \notin \mathcal{S}(\hat{\mathbf{x}}_i)$ is never predicted. This can cause a catastrophic collapse of the model likelihood if even a single positive label fails to be shortlisted by the shortlister, i.e. if the shortlister admits even a single false negative. To address this, and allow DECAF to continue working with shortlisters with high but still imperfect recall, we augment the likelihood model as follows

$$\mathbb{P}[y_{il} \mid \hat{\mathbf{x}}_i, \mathbf{w}_l] = \begin{cases} (1 + \exp(y_{il} \cdot \langle \hat{\mathbf{x}}_i, \mathbf{w}_l \rangle))^{-1} & l \in \mathcal{S}(\hat{\mathbf{x}}_i) \\ y_{il}\left(\eta - \frac{1}{2}\right) + \frac{1}{2} & l \notin \mathcal{S}(\hat{\mathbf{x}}_i) \end{cases},$$

where $\eta \in (0, 1]$ is some default likelihood value assigned to positive labels that escape shortlisting (recall that $y_{il} \in \{-1, +1\}$). Essentially, for non-shortlisted labels, we posit their probability of being relevant as η. The value of η will be tuned later.

Note that we must set $\eta \ll 1$ so as to ensure that these default likelihood scores do not interfere with the prediction pipeline which discards non-shortlisted labels. We will see that our calculations do result in an extremely small value of η as the optimal value. However, also note that we cannot simply set $\eta = 0$ since that would lead to a catastrophic collapse of the model likelihood to zero if the shortlister has even one false negative. Although our shortlister does offer good recall even with shortists of small length (e.g. 85% with a shortlist of length ≈ 200), demanding 100% recall would require exorbitantly large beam sizes that would slow down prediction greatly. Thus, it is imperative that the augmented likelihood model itself account for shortlister failures.

To incorporate the above augmentation, we also redefine our log-likelihood score function to handle document-label pairs $(i, l) \in [N] \times [L]$ such that $l \notin \mathcal{S}(\hat{\mathbf{x}}_i)$

$$\ell_{il}(\Theta \mid \mathcal{S}) = \begin{cases} \ln(1 + \exp(y_{il} \cdot \langle \hat{\mathbf{x}}_i, \mathbf{w}_l \rangle)) & l \in \mathcal{S}(\hat{\mathbf{x}}_i) \\ -\ln\left(y_{il}\left(\eta - \frac{1}{2}\right) + \frac{1}{2}\right) & l \notin \mathcal{S}(\hat{\mathbf{x}}_i) \end{cases},$$

Note the negative sign in the second case since ℓ_{ij} is the negative log-likelihood expression. We will also benefit from defining the following *residual* loss term

$$\Delta(\Theta \mid \mathcal{S}) = \sum_{i \in [N]} \sum_{l \notin \mathcal{S}(\hat{\mathbf{x}}_i)} \ell_{il}(\Theta)$$

Note that Δ simply sums up loss terms corresponding to all labels omitted by the shortlister. We will establish the result claimed in the theorem by comparing the performance offered by $\hat{\Theta}$ and Θ^* on the loss terms given by $\tilde{\mathcal{L}}$ and Δ. Note that for any Θ we always have the following decomposition

$$\mathcal{L}(\Theta) = \frac{1}{NL}\left(\frac{NLB}{K}\cdot\tilde{\mathcal{L}}(\Theta\,|\,\mathcal{S}) + \Delta(\Theta\,|\,\mathcal{S}\right)$$

Now, since $\hat{\Theta}$ optimizes $\tilde{\mathcal{L}}$, we have $\tilde{\mathcal{L}}(\hat{\Theta}\,|\,\mathcal{S}) \leq \tilde{\mathcal{L}}(\Theta^*\,|\,\mathcal{S})$ which settles the first term in the above decomposition. To settle the second term, we note that as per the recall r and label sparsity s terms defined in the statement of the theorem, the number of positive labels not shortlisted by the shortlister $\mathcal{S}$ throughout the dataset is $\text{FPR}\cdot NL$ where $\text{FPR} = (1-r)s$ is the false negative rate of the shortlister. Similarly, the number of negative labels not shortlisted by the shortlister throughout the dataset by $(L-B)N$ can be seen to be $\text{TNR}\cdot NL$ where $\text{TNR} = \left((1-s) - \frac{B}{K} + rs\right)$ is the true negative rate of the shortlister. This gives us

$$\Delta(\hat{\Theta}\,|\,\mathcal{S}) = \left(\text{FPR}\cdot\ln\frac{1}{\eta} + \text{TNR}\cdot\ln\frac{1}{1-\eta}\right)\cdot NL$$

It is easy to see that the optimal value of η for the above expression is $\eta = \frac{\text{FPR}}{\text{FPR+TNR}}$. For example, in the LF-WikiSeeAlsoTitles-320K dataset, which has $s \approx 6.75\times10^{-6}, r \approx 0.85, B = 160, K = 2^{17}$, this gives a value of $\text{FPR} \approx 1.01\times10^{-6}, \text{TNR} \approx 0.999$ which gives $\eta \approx 1.01\times10^{-6}$. This confirms that the augmentation indeed does not interfere with the prediction pipeline and labels not shortlisted can be safely ignored. However, moving on and plugging this optimal value of η into the expression tells us that

$$\Delta(\hat{\Theta}\,|\,\mathcal{S}) = \left(\frac{\text{FPR}}{\text{TNR}}\ln\left(1+\frac{\text{TNR}}{\text{FPR}}\right) + \ln\left(1+\frac{\text{FPR}}{\text{TNR}}\right)\right)\cdot NL.$$

Since TNR → 1 (for example, we saw $\text{TNR} \approx 0.999$ above), we simplify this to $\frac{\text{FPR}}{\text{TNR}} = \mathcal{O}\left(\text{FPR}\right)$ and use the inequality $\ln(1+v) \leq v$ for all $v > 0$ to conclude that $\Delta(\hat{\Theta}\,|\,\mathcal{S}) \leq \mathcal{O}\left(\text{FPR}\ln\frac{1}{\text{FPR}} + \text{FPR}\right) = \mathcal{O}\left(s(1-r)\ln\left(\frac{1}{s(1-r)}\right)\right)$. Using $\Delta(\Theta^*\,|\,\mathcal{S}) \geq 0$ settles the second term in the decomposition by establishing that $\Delta(\hat{\Theta}\,|\,\mathcal{S}) - \Delta(\Theta^*\,|\,\mathcal{S}) \leq \mathcal{O}\left(s(1-r)\ln\left(\frac{1}{s(1-r)}\right)\right)\cdot NL$. Combining the two terms in the decomposition above gives us

$$\mathcal{L}(\hat{\Theta}) - \mathcal{L}(\Theta^*) = \frac{1}{NL}\left(\frac{NLB}{K}\cdot(\tilde{\mathcal{L}}(\Theta\,|\,\mathcal{S}) - \tilde{\mathcal{L}}(\Theta^*\,|\,\mathcal{S})) + (\Delta(\Theta\,|\,\mathcal{S}) - \Delta(\Theta^*\,|\,\mathcal{S}))\right) \leq \mathcal{O}\left(s(1-r)\ln\left(\frac{1}{s(1-r)}\right)\right),$$

which finishes the proof of the theorem.

We conclude this discussion by noting that since $\mathcal{L}$ and $\tilde{\mathcal{L}}$ are non-convex objectives due to the non-linear architecture encoded by the model parameters Θ, it may not be able to solve these objectives optimally in practice. Thus, in practice, all we may be ensure is that

$$\tilde{\mathcal{L}}(\hat{\Theta}\,|\,\mathcal{S}) \leq \min_{\Theta}\tilde{\mathcal{L}}(\Theta\,|\,\mathcal{S}) + \epsilon_{\text{opt}}$$

where ϵ_{opt} is the sub-optimality in optimizing the objective $\tilde{\mathcal{L}}$ due to factors such as sub-optimal initialization, training, premature termination, etc. It is easy to see that the main result of the theorem continues to hold since we now have $\tilde{\mathcal{L}}(\hat{\Theta}\,|\,\mathcal{S}) \leq \tilde{\mathcal{L}}(\Theta^*\,|\,\mathcal{S}) + \epsilon_{\text{opt}}$ which gives us the modified result as follows

$$\mathcal{L}(\hat{\Theta}) \leq \min_{\Theta}\mathcal{L}(\Theta) + \mathcal{O}\left(s(1-r)\ln\left(\frac{1}{s(1-r)}\right)\right) + \frac{B}{K}\epsilon_{\text{opt}}.$$

A.2 Time Complexity Analysis for DECAF

In this section, we discuss the time complexity of the various modules in DECAF, as well as derive the prediction and training complexities.

Notation: Recall from Section 3 that DECAF learns D-dimensional representations for all V tokens ($\mathbf{e}_t, t \in [V]$), that are used to create embeddings for all L labels $\hat{\mathbf{z}}_l^1, l \in [L]$, and all N training documents $\hat{\mathbf{x}}_i, i \in [N]$. We introduce some additional notation to facilitate the discussion: we use $\hat{V}_x$ to denote the average number of unique tokens present in a document i.e. $\hat{V}_x = \frac{1}{N}\sum_{i=1}^N \|\mathbf{x}_i\|_0$ where $\|\cdot\|_0$ is the sparsity "norm" that gives the number of non-zero elements in a vector. We similarly use $\hat{V}_y = \frac{1}{L}\sum_{l=1}^L \|\mathbf{z}_l\|_0$ to denote the average number of tokens in a label text. Let $\hat{L} = \frac{1}{N}\sum_{i=1}^N \|\mathbf{y}_i\|_0$ denote the average number of labels per document and also let $\hat{N} = \frac{N\hat{L}}{L}$ denote the average number of documents per label. We also let M denote the mini-batch size (DECAF used $M = 255$ for all datasets – see Table 8).

Embedding Block: Given a text with $\hat{V}$ tokens, the embedding block requires $\hat{V}D$ operations to aggregate token embeddings and $D^2 + 3D$ operations to execute the residual block and the combination block, for a total of $\mathcal{O}\left(\hat{V}D + D^2\right)$ operations. Thus, to encode a label (respectively document) text, it takes $\mathcal{O}\left(\hat{V}_y D + D^2\right)$ (respectively $\mathcal{O}\left(\hat{V}_x D + D^2\right)$) operations on average.

Prediction: Given a test document, assuming that it contain $\hat{V}_x$ tokens, embedding takes $\mathcal{O}\left(\hat{V}_x D + D^2\right)$ operations, executing the shortlister by identifying the top B clusters takes $\mathcal{O}\left(KD + K\log K\right)$ operations. These clusters contain a total of $\frac{LB}{K}$ labels. The ranker takes $\mathcal{O}\left(\frac{LB}{K}D + \frac{LB}{K}\log\left(\frac{LB}{K}\right)\right)$ operations to execute the $\frac{LB}{K}$ OvA linear models corresponding to these shortlisted labels to obtain the top-ranked predictions. Thus, prediction takes $\mathcal{O}\left(\hat{V}_x D + D^2 + KD + K\log K\right) = \mathcal{O}\left(KD\right)$ time since usually $\frac{LB}{K} \leq K, \hat{V}_x \leq K$ and $\log K \leq D \leq K$.

Module I Training: Creation of all L label centroids $\mathbf{c}_l$ takes $O\left(L\hat{N}\hat{V}_x\right)$ time. These centroids are $O\left(\hat{N}\hat{V}_x\right)$-sparse on average. Clustering these labels using hierarchical balanced binary clustering for $\log K$ levels to get K balanced clusters takes time $O\left(L\hat{N}\hat{V}_x \log K\right)$. Computing meta label text representations $\mathbf{u}_m$ for all meta labels takes $O\left(L\hat{V}_y\right)$ time. The vectors $\mathbf{u}_m$ are $\frac{\hat{V}_y L}{K}$-sparse on average. To compute the complexity of learning the K OvA meta-classifiers, we calculate below the cost of a single back-propagation step when using a mini-batch of size M. Computing the document and meta-label features of all M documents in the mini-batch and K meta-labels takes on average $O\left((D^2 + \hat{V}_x D)M\right)$ and $O\left(\left(D^2 + \frac{\hat{V}_y L}{K} \cdot D\right)K\right)$ time respectively. Computing the scores for all the OvA meta classifiers for all documents in the mini-batch takes $O\left(MKD\right)$ time. Overestimating that the K meta label texts together cover all V tokens, updating the residual layer parameters $\mathbf{R}$, the combination block parameters, and the token embeddings $\mathbf{E}$ using back-propagation takes at most $O\left((D^2+V)MK\right)$ time.

Module II Training: Recreating all L label centroids $\mathbf{c}_l$ now takes $O\left(L\hat{N}\hat{V}_x D\right)$ time. Clustering the labels takes time $O\left(LD\log K\right)$. Computing document features in a mini-batch of size M takes $O\left((\hat{V}_x D + D^2)M\right)$ time as before. Computing the meta-label representations $\hat{\mathbf{u}}_m^1$ for all K meta-labels now takes $O\left((\hat{V}_y D + D^2)L\right)$ time. Computing the scores for all the OvA meta classifiers for all documents in the mini-batch takes $O\left(MKD\right)$ time as before. Next, updating the model parameters as well as the refinement vectors $\hat{\mathbf{u}}_m^2, m \in [K]$ takes at most $O\left((D^2+V)MK\right)$ time time as before. The added task of updating $\hat{\mathbf{u}}_m^2$ does not affect the asymptotic complexity of this module. Generating the shortlists for all N training points is essentially a prediction step and takes $O\left(NKD\right)$ time.

Module II Initializations: Model parameter initializations take $O\left(D^2\right)$ time. Initializing the refinement vectors $\hat{\mathbf{z}}_l^2$ takes $O\left(L\hat{V}_y D\right)$ time.

Module IV Training: Given the shortlist of LB/K labels per training point generated in Module II, training the OvA classifiers by fine-tuning the model parameters and learning the refinement vectors $\hat{\mathbf{z}}_l^2, l \in [L]$ is made much less expensive than $O\left(NLD\right)$. Computing document features in a mini-batch of size M takes $O\left((\hat{V}_x D + D^2)M\right)$ time as before. However, label representations $\hat{\mathbf{z}}_l^1$ of only shortlisted labels need be computed. Since there are atmost $\left(\frac{LB}{K} + \hat{L}\right)M$ of them (accounting for hard negatives and all positives), this takes $O\left((\hat{V}_y D + D^2)M\left(\frac{LB}{K}+\hat{L}\right)\right)$ time. Next, updating the model parameters as well as the refinement vectors $\hat{\mathbf{z}}_l^2$ for shortlisted takes at most $O\left((D^2 + (\hat{V}_x + \hat{V}_y)D)M\left(\frac{LB}{K}+\hat{L}\right)\right)$ time. This can be simplified to $O\left(M\left(\frac{LB}{K}+\hat{L}\right)D^2\right) = O\left(MD^2\log^2 L\right)$ time per mini-batch since $\hat{V}_x, \hat{V}_y \le D$, usually $\hat{L} \le O\left(\log L\right)$ and DECAF chooses $\frac{B}{K} \le O\left(\frac{\log^2 L}{L}\right)$ for large datasets such as LF-AmazonTitles-1.3M and LF-P2PTitles-2M (see Table 8), thus ensuring an OvA training time that scales at most as $\log^2 L$ with the number of labels.

A.3 Dataset Preparation and Evaluation Details

Train-test splits were generated using a random 70:30 split keeping only those labels that have at least 1 test as well as 1 train point. For sake of validation, 5% of training data points were randomly sampled.

Reciprocal pair removal: It was observed that in certain datasets, documents were mapped to themselves. For instance, the product with title "Dinosaur" was tagged with the label "Dinosaur" itself in the LF-AmazonTitles-131K dataset. Algorithms could achieve disproportionately high P@1 by making such trivial predictions without learning anything useful. Additionally, in product-to-product and related webpage recommendation tasks, both documents and labels come from the same set/universe. This allows for *reciprocal pairs* to exist where a data point has document A and label B in its ground truth but a separate data point has document B and label A in its ground truth. We affectionately call these AB and BA pairs respectively. If these pairs are split across train and test sets, an algorithm could simply memorize the AB pair while training and predict the BA pair during testing to achieve very high P@1. Moreover, such predictions did not add to the quality of predictions in real-life applications. Hence, methods were not rewarded for making such trivial predictions. Table 1 reports numbers as per this very evaluation strategy. Additionally, coverage (C@20) is reported in Table 2 to verify that prediction accuracy is not being achieved at the expense of label coverage.

A.4 Evaluation metrics

Performance was evaluated using precision@k and nDCG@k metrics. Performance was also evaluated using propensity scored metrics, namely propensity scored precision@k and nDCG@k (with k = 1, 3 and 5) for extreme classification. The propensity scoring model and values available on The Extreme Classification Repository [4] were used for the publicly available datasets. For the proprietary datasets, the method outlined in [15] was used. For a predicted score vector $\hat{\mathbf{y}} \in R^L$ and ground truth label vector $\mathbf{y} \in \{0,1\}^L$, the metrics are defined below. In the following, p_l is propensity score of the label l as proposed in [15].

$$P@k = \frac{1}{k} \sum_{l \in rank_k(\hat{\mathbf{y}})} y_l \qquad\qquad PSP@k = \frac{1}{k} \sum_{l \in rank_k(\hat{\mathbf{y}})} \frac{y_l}{p_l}$$

$$DCG@k = \frac{1}{k}\sum_{l\in rank_k(\hat{\mathbf{y}})} \frac{y_l}{\log(l+1)} \qquad PSDCG@k = \frac{1}{k}\sum_{l\in rank_k(\hat{\mathbf{y}})} \frac{y_l}{p_l \log(l+1)}$$

$$nDCG@k = \frac{DCG@k}{\sum_{l=1}^{\min(k,||\mathbf{y}||_0)} \frac{1}{\log(l+1)}} \qquad PSnDCG@k = \frac{PSDCG@k}{\sum_{l=1}^{k} \frac{1}{\log l+1}},$$

A.5 Further Details about Experiments and Ablation Studies

Recap of Notation: Let us recall from Section 3, that L denotes the number of labels and V denotes the total number of tokens appearing across label and document texts. The training set of N documents is presented as $\left\{(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^{N}\right\}$ with each document represented as a bag of tokens $\mathbf{x}_i \in \mathbb{R}^V$ with x_{it} representing the TF-IDF weight of token $t \in [V]$ in the i^{th} document, and the ground truth label vector $\mathbf{y}_i \in \{-1, +1\}^L$ such that $y_{il} = +1$ if label $l \in [L]$ is relevant to document i and $y_{il} = -1$ otherwise. For each label $l \in [L]$, its label text is similarly represented as a bag of TF-IDF scores $\mathbf{z}_l \in \mathbb{R}^V$. DECAF learns D-dimensional embeddings for tokens, documents as well as labels.

Incorporating Label text into existing BoW XML methods: XML classifiers such as Parabel, DiSMEC, Bonsai, *etc*, use a fixed BoW (bag-of-words)-based representation of documents to learn their classifiers. Label text was incorporated into these classifiers as follows: for every document i, let $s_{il} \in \mathbb{R}$ be the relevance score the XML classifier predicted for label l for document i. We augmented this score to incorporate label text by computing $\tilde{s}_{il} = \alpha \cdot s_{il} + (1-\alpha)\sigma\left(\langle \mathbf{x}_i, \mathbf{z}_l\rangle\right)$. Here, $\alpha \in [0, 1]$ was fine tuned to offer the best results. Table 4 shows that incorporating label text, even in this relatively crude way, still benefits accuracy.

Generating alternative shortlists for DECAF: DECAF learns a shortlister to generate a subset of labels with high recall, from an extremely large output space. Experiments were also conducted to use existing scalable XML algorithms *e.g.* Parabel or ANNS data structures *e.g.* HNSW as possible alternatives to generating this shortlist. Label centroids using learnt intermediate feature representations were provided to Parabel and HNSW in order to partition the label space. However, as Table 5 shows, this leads to significant reduction in precision as well as recall (upto 2%) which adversely impacted the performance of the final ranking by DECAF.

Varying the shortlister fan-out in DECAF: DECAF uses Modules I and II to learn a shortlister. In Module I, DECAF clusters the extremely large label space (in millions) to a smaller number of $K = 2^{17} \approx 130K$ meta-labels. In Module II, DECAF fine-tunes the re-ranker to generate a shortlist of labels. For details of training please refer to section 3 in the main paper. Experiments were conducted to observe the impact of the fan-out K. In particular fan-out was restricted to $2^{13} \approx 8K$ which is also a value used by contemporary algorithms such as AttentionXML and the X-Transformer. It was observed that to maintain a high recall (of around 85%) during training DECAF had to increase the beam-size by 2× which leads to increase in training time as well as a drop in accuracy (see Table 6 DECAF-8K). AttentionXML and X-Transformer were found to be computationally expensive and could not be scaled to use 2^{17} clusters to check whether increasing fan-out benefits them as it does DECAF.

Varying the label classifier components in DECAF: As outlined in Section 3, DECAF makes crucial use of label text embeddings while learning its label classifiers $\mathbf{w}_l, l \in [L]$, with two components for each label l a) $\hat{\mathbf{z}}_l^1$ that is simply the label text embedding, and b) $\hat{\mathbf{z}}_l^2$ that is a refinement vector. $\hat{\mathbf{z}}_l^2$ was initialized with $\mathbf{E}\mathbf{z}_l$ and then fine-tuned jointly with other model parameters such as those within the residual and combination blocks, etc. An experiment was conducted in which the label embedding component $\hat{\mathbf{z}}_l^1$ was removed from the label classifier (effectively done by setting $\hat{\mathbf{z}}_l^1 = \mathbf{0}, \forall l \in [L]$) and $\hat{\mathbf{z}}_l^2$ was randomly initialized instead. We call this configuration DECAF-$\hat{\mathbf{z}}^2$ (see Table 6). Another experimented was conducted to understand the importance of the refinement vector $\hat{\mathbf{z}}_l^2$. In this experiment, $\hat{\mathbf{z}}_l^2$ was explicitly set to $\mathbf{0}$ and we used $\mathbf{w}_l = \hat{\mathbf{z}}_l^1$. We call this configuration DECAF-$\hat{\mathbf{z}}^1$ (see Table 6).DECAF was found to be upto 5% more accurate as compared to these variants. These experiments suggest that the novel combination of two label classifier components as proposed by DECAF, namely $\hat{\mathbf{z}}_l^1$ and $\hat{\mathbf{z}}_l^2$ is essential for achieving high accuracy.

Please go to the next page for dataset statistics and hyperparameter details.

Table 7: Dataset Statistics. A ‡ sign denotes information that was redacted for the proprietary datasets. The first four rows are public short-text datasets. The next three rows are public full-text versions of the first three rows. The last two rows are proprietary short-text datasets. Dataset names with an asterisk * next to them correspond to product-to-category tasks whereas others correspond to product-to-product tasks.

Dataset	Train Documents	Labels	Tokens	Test Instances	Average Labels	Average Points	Average Tokens	Average Tokens
	N	L	V	N'	per Document	per label	per Document	per Label
Short text dataset statistics								
LF-AmazonTitles-131K	294,805	131,073	40,000	134,835	2.29	5.15	7.46	7.15
LF-WikiSeeAlsoTitles-320K	693,082	312,330	40,000	177,515	2.11	4.68	3.97	3.92
LF-WikiTitles-500K*	1,813,391	501,070	80,000	783,743	4.74	17.15	3.72	4.16
LF-AmazonTitles-1.3M	2,248,619	1,305,265	128,000	970,237	22.20	38.24	9.00	9.45
Long text dataset statistics								
LF-Amazon-131K	294,805	131,073	80,000	134,835	2.29	5.15	64.28	4.87
LF-WikiSeeAlso-320K	693,082	312,330	200,000	177,515	2.11	4.67	99.79	2.68
LF-Wikipedia-500K*	1,813,391	501,070	500,000	783,743	4.74	17.15	165.18	3.24
Proprietary dataset								
LF-P2PTitles-300K	1,366,429	300,000	‡	585,602	‡	‡	‡	‡
LF-P2PTitles-2M	2,539,009	1,640,898	‡	1,088,146	‡	‡	‡	‡

Table 8: Parameter settings for DECAF on different datasets. Apart from the hyperparameters mentioned in the table below, all other hyperparameters were held constant across datasets. All ReLU layers were followed by a dropout layer with 50% drop-rate in Module-I and 20% for the rest of the modules. Learning rate was decayed by a decay factor of 0.5 after interval 0.5× **epoch length. Batch size was taken to be 255 for all datasets. Module I used 20 epochs with initial learning rate of 0.01. In Module II, 10 epochs were used with an initial learning rate of 0.008 for all datasets.**

Dataset	Beam Size	Embedding Dimension	Cluster Size
LF-AmazonTitles-131K	200	300	2^{15}
LF-WikiSeeAlsoTitles-320K	160	300	2^{17}
LF-AmazonTitles-1.3M	100	512	2^{17}
LF-Amazon-131K	200	512	2^{15}
LF-WikiSeeAlso-320K	160	512	2^{17}
LF-P2PTitles-300K	160	300	2^{17}
LF-P2PTitles-2M	40	512	2^{17}
LF-WikiTitles-500K	100	512	2^{17}
LF-Wikipedia-500K	100	512	2^{17}

Please go to the next page for detailed experimental results.

Table 9: A comparison of DECAF on publicly available product-to-product datasets. The first 3 rows are short-text datasets whereas the last two rows are long-text versions of the first two. DECAF offers predictions that are the most accurate based on all evaluation metrics, and an order of magnitude faster as compared to existing deep learning based approaches. Methods marked with a '-' sign could not be scaled for the given dataset within the available resources.

Dataset	Method	P@1	P@3	P@5	N@3	N@5	PSP@1	PSP@3	PSP@5	PSN@3	PSN@5	Model Size (GB)	Training Time (hr)	Prediction Time (ms)
LF-AmazonTitles-131K	DECAF	**38.4**	**25.84**	**18.65**	**39.43**	**41.46**	**30.85**	**36.44**	**41.42**	**34.69**	**37.13**	0.81	2.16	0.1
	Astec	37.12	25.2	18.24	38.17	40.16	29.22	34.64	39.49	32.73	35.03	3.24	1.83	2.34
	AttentionXML	32.25	21.7	15.61	32.83	34.42	23.97	28.6	32.57	26.88	28.75	2.61	20.73	5.19
	MACH	33.49	22.71	16.45	34.36	36.16	24.97	30.23	34.72	28.41	30.54	2.35	3.3	0.23
	X-Transformer	29.95	18.73	13.07	28.75	29.6	21.72	24.42	27.09	23.18	24.39	-	-	15.38
	Siamese	13.81	8.53	5.81	13.32	13.64	13.3	12.68	13.36	12.69	13.06	0.6	6.92	0.2
	Parabel	32.6	21.8	15.61	32.96	34.47	23.27	28.21	32.14	26.36	28.21	0.34	0.03	0.69
	Bonsai	34.11	23.06	16.63	34.81	36.57	24.75	30.35	34.86	28.32	30.47	0.24	0.1	7.49
	DiSMEC	35.14	23.88	17.24	36.17	38.06	25.86	32.11	36.97	30.09	32.47	0.11	3.1	5.53
	PfastreXML	32.56	22.25	16.05	33.62	35.26	26.81	30.61	34.24	29.02	30.67	3.02	0.26	2.32
	XT	31.41	21.39	15.48	32.17	33.86	22.37	27.51	31.64	25.58	27.52	0.84	9.46	9.12
	Slice	30.43	20.5	14.84	31.07	32.76	23.08	27.74	31.89	26.11	28.13	0.39	0.08	1.58
	AnneXML	30.05	21.25	16.02	31.58	34.05	19.23	26.09	32.26	23.64	26.6	1.95	0.08	0.11
LF-WikiSeeAlsoTitles-320K	DECAF	**25.14**	**16.9**	**12.86**	**24.99**	**25.95**	**16.73**	**18.99**	**21.01**	**19.18**	**20.75**	1.76	11.16	0.09
	Astec	22.72	15.12	11.43	22.16	22.87	13.69	15.81	17.5	15.56	16.75	7.3	4.17	2.67
	AttentionXML	17.56	11.34	8.52	16.58	17.07	9.45	10.63	11.73	10.45	11.24	6.02	56.12	7.08
	MACH	18.06	11.91	8.99	17.57	18.17	9.68	11.28	12.53	11.19	12.14	2.51	8.23	0.52
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Siamese	10.69	6.28	4.51	9.79	9.91	10.1	9.43	9.59	10.22	10.47	0.67	11.58	0.17
	Parabel	17.68	11.48	8.59	16.96	17.44	9.24	10.65	11.8	10.49	11.32	0.6	0.07	0.8
	Bonsai	19.31	12.71	9.55	18.74	19.32	10.69	12.44	13.79	12.29	13.29	0.37	0.37	14.82
	DiSMEC	19.12	12.93	9.87	18.93	19.71	10.56	13.01	14.82	12.7	14.02	0.19	15.56	11.02
	PfastreXML	17.1	11.13	8.35	16.8	17.35	12.15	12.51	13.26	12.81	13.48	6.77	0.59	2.59
	XT	17.04	11.31	8.6	16.61	17.24	8.99	10.52	11.82	10.33	11.26	-	5.28	12.86
	Slice	18.55	12.62	9.68	18.29	19.07	11.24	13.45	15.2	13.03	14.23	0.94	0.2	1.85
	AnneXML	16.3	11.24	8.84	16.19	17.14	7.24	9.63	11.75	9.06	10.43	4.22	0.21	0.13
LF-AmazonTitles-1.3M	DECAF	**50.67**	**44.49**	**40.35**	**48.05**	**46.85**	22.07	26.54	29.3	25.06	26.85	9.62	74.47	0.16
	Astec	48.82	42.62	38.44	46.11	44.8	21.47	25.41	27.86	24.08	25.66	26.66	18.54	2.61
	AttentionXML	45.04	39.71	36.25	42.42	41.23	15.97	19.9	22.54	18.23	19.6	28.84	380.02	29.53
	MACH	35.68	31.22	28.35	33.42	32.27	9.32	11.65	13.26	10.79	11.65	7.68	60.39	2.09
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Siamese	-	-	-	-	-	-	-	-	-	-	-	-	-
	Parabel	46.79	41.36	37.65	44.39	43.25	16.94	21.31	24.13	19.7	21.34	11.75	1.5	0.89
	Bonsai	47.87	42.19	38.34	45.47	44.35	18.48	23.06	25.95	21.52	23.33	9.02	7.89	39.03
	DiSMEC	-	-	-	-	-	-	-	-	-	-	-	-	-
	PfastreXML	37.08	33.77	31.43	36.61	36.61	**28.71**	**30.98**	**32.51**	**29.92**	**30.73**	29.59	9.55	23.64
	XT	40.6	35.74	32.01	38.18	36.68	13.67	17.11	19.06	15.64	16.65	7.9	82.18	5.94
	Slice	34.8	30.58	27.71	32.72	31.69	13.8	16.87	18.89	15.62	16.74	5.98	0.79	1.45
	AnneXML	47.79	41.65	36.91	44.83	42.93	15.42	19.67	21.91	18.05	19.36	14.53	2.48	0.12
LF-Amazon-131K	DECAF	**42.94**	28.79	**21**	**44.25**	**46.84**	**34.52**	**41.14**	**47.33**	**39.35**	**42.48**	1.86	1.8	0.1
	Astec	-	-	-	-	-	-	-	-	-	-	-	-	-
	AttentionXML	42.9	**28.96**	20.97	44.07	46.44	32.92	39.51	45.24	37.49	40.33	5.04	50.17	12.33
	MACH	34.52	23.39	17	35.53	37.51	25.27	30.71	35.42	29.02	31.33	4.57	13.91	0.25
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Bonsai	40.23	27.29	19.87	41.46	43.84	29.6	36.52	42.39	34.43	37.34	0.46	0.4	7.41
	DiSMEC	41.68	28.32	20.58	43.22	45.69	31.61	38.96	45.07	36.97	40.05	0.45	7.12	15.48
	PfastreXML	35.83	24.35	17.6	36.97	38.85	28.99	33.24	37.4	31.65	33.62	0.01	1.54	3.32
	XT	34.31	23.27	16.99	35.18	37.26	24.35	29.81	34.7	27.95	30.34	0.92	1.38	7.42
	Slice	32.07	22.21	16.52	33.54	35.98	23.14	29.08	34.63	27.25	30.06	0.39	0.11	1.35
	AnneXML	35.73	25.46	19.41	37.81	41.08	23.56	31.97	39.95	29.07	33	4.01	0.68	0.11
LF-WikiSeeAlso-320K	DECAF	**41.36**	**28.04**	**21.38**	**41.55**	**43.32**	**25.72**	**30.93**	**34.89**	**30.69**	**33.69**	4.84	13.4	0.09
	Astec	-	-	-	-	-	-	-	-	-	-	-	-	-
	AttentionXML	40.5	26.43	19.87	39.13	40.26	22.67	26.66	29.83	26.13	28.38	7.12	90.37	12.6
	MACH	27.18	17.38	12.89	26.09	26.8	13.11	15.28	16.93	15.17	16.48	11.41	50.22	0.54
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Bonsai	34.86	23.21	17.66	34.09	35.32	18.19	22.35	25.66	21.62	23.84	0.84	1.39	8.94
	DiSMEC	34.59	23.58	18.26	34.43	36.11	18.95	23.92	27.9	23.04	25.76	1.28	58.79	75.52
	PfastreXML	28.79	18.38	13.6	27.69	28.28	17.12	18.19	19.43	18.23	19.2	14.02	4.97	2.68
	XT	30.1	19.6	14.92	28.65	29.58	14.43	17.13	19.69	16.37	17.97	2.2	3.27	4.79
	Slice	27.74	19.39	15.47	27.84	29.65	13.07	17.5	21.55	16.36	18.9	0.94	0.2	1.18
	AnneXML	30.79	20.88	16.47	30.02	31.64	13.48	17.92	22.21	16.52	19.08	12.13	2.4	0.11

Table 10: A comparison of DECAF's performance on product-to-category datasets. The first row is a short-text dataset and the second row its long-text counterpart. Although DECAF focuses on product-to-product tasks, it is nevertheless competitive in terms of accuracy, as well as an order of magnitude faster in prediction as compared to leading deep learning approaches. Methods marked with a '-' sign could not be scaled for the given dataset within the available resources. The AttentionXML method used a non-standard version of the Wikipedia-500K dataset. All other methods, including DECAF, used the standard version of the dataset.

Dataset	Method	P@1	P@3	P@5	N@3	N@5	PSP@1	PSP@3	PSP@5	PSN@3	PSN@5	Model Size (GB)	Training Time (hrs)	Prediction Time (ms)
LF-WikiTitles-500K	DECAF	44.21	24.64	17.36	**33.55**	**31.92**	**19.29**	**19.82**	**19.96**	**21.26**	**22.95**	4.53	42.26	0.09
	Astec-3	**44.4**	**24.69**	**17.49**	33.43	31.72	18.31	18.25	18.56	19.57	21.09	15.01	13.5	2.7
	AttentionXML	40.9	21.55	15.05	29.38	27.45	14.8	13.97	13.88	15.24	16.22	14.01	133.94	9
	MACH	37.74	19.11	13.26	26.63	24.94	13.71	12.14	12	13.63	14.54	4.73	22.46	0.8
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Siamese	-	-	-	-	-	-	-	-	-	-	-	-	-
	Parabel	40.41	21.98	15.42	29.89	28.15	15.55	15.32	15.35	16.5	17.66	2.7	0.42	0.81
	Bonsai	40.97	22.3	15.66	30.35	28.65	16.58	16.34	16.4	17.6	18.85	1.63	2.03	17.38
	DiSMEC	39.42	21.1	14.85	28.87	27.29	15.88	15.54	15.89	16.76	18.13	0.68	48.27	11.71
	PfastreXML	35.71	19.27	13.64	26.45	25.15	18.23	15.42	15.08	17.34	18.24	20.41	3.79	9.37
	XT	38.19	20.74	14.68	28.15	26.64	14.2	14.14	14.41	15.18	16.45	3.1	8.78	7.56
	Slice	25.48	15.06	10.98	20.67	20.52	13.9	13.33	13.82	14.5	15.9	2.3	0.74	1.76
	AnneXML	39	20.66	14.55	28.4	26.8	13.91	13.38	13.75	14.63	15.88	11.18	1.98	0.13
LF-Wikipedia-500K	DECAF	73.96	54.17	42.43	66.31	64.81	32.13	40.13	44.59	39.57	43.7	9.34	44.23	0.09
	Astec	-	-	-	-	-	-	-	-	-	-	-	-	-
	AttentionXML	**82.73**	**63.75**	**50.41**	**76.56**	**74.86**	**34**	**44.32**	**50.15**	**42.99**	**47.69**	9.73	221.6	12.38
	MACH	52.48	31.93	23.34	41.7	39.43	17.92	18.16	18.66	19.45	20.77	28.12	220.07	0.82
	X-Transformer	-	-	-	-	-	-	-	-	-	-	-	-	-
	Siamese	-	-	-	-	-	-	-	-	-	-	-	0.03	-
	Parabel	70.14	50.62	39.45	61.86	59.89	27.25	32.52	35.93	32.29	35.31	5.51	3.02	2.01
	Bonsai	70.56	51.11	39.86	62.47	60.61	28.18	33.86	37.55	33.58	36.86	3.94	17.22	22.23
	DiSMEC	-	-	-	-	-	-	-	-	-	-	-	-	-
	PfastreXML	61.24	41.59	31.75	52.26	50.34	33.3	32.56	33.67	33.77	35.25	48.26	24.71	7.69
	XT	66.98	48.33	37.82	58.94	57.19	24.78	30.06	33.46	29.63	32.51	3.9	16.73	3.81
	Slice	47.51	32.34	25.07	40.56	39.51	19.6	21.99	24.6	22.2	24.53	2.3	0.67	1.58
	AnneXML	64.77	43.24	32.79	54.63	52.51	24.08	28.25	31.2	28.47	31.3	49.25	14.97	5.15

Table 11: A comparison of DECAF's performance on the proprietary datasets. DECAF can be an order of magnitude faster in prediction as compared to existing deep learning approaches.

Dataset	Method	P@1	P@3	P@5	N@3	N@5	PSP@1	PSP@3	PSP@5	PSN@3	PSN@5
P2PTitles-300K	DECAF	**47.17**	**30.67**	**22.69**	**53.62**	**57.06**	**42.43**	**55.07**	**62.3**	**49.86**	**53.27**
	Astec	44.3	28.95	21.56	50.36	53.67	39.44	50.9	57.83	45.99	49.12
	Parabel	43.14	28.34	20.99	48.73	51.75	37.26	48.87	55.32	43.45	46.32
	PfastreXML	39.4	25.6	18.77	44.59	46.98	35.79	45.13	49.9	40.98	43.03
	Slice	31.27	28.91	**25.19**	31.5	33.2	27.03	30.44	34.95	28.54	30.77
P2PTitles-2M	DECAF	**40.27**	**36.65**	**31.45**	**40.4**	**42.49**	**36.65**	**40.14**	**45.15**	**38.23**	**40.99**
	Astec	36.34	33.33	28.74	36.63	38.63	32.75	36.3	41	34.43	36.97
	Parabel	35.26	32.44	28.06	35.3	36.89	30.21	33.85	38.46	31.63	33.71
	PfastreXML	30.52	28.68	24.6	31.5	33.23	28.84	32.1	35.65	30.56	32.52
	Slice	31.27	28.91	25.19	31.5	33.2	27.03	30.44	34.95	28.54	30.77

Table 12: A subjective comparison of DECAF's prediction quality as compared to state-of-the-art deep learning, as well as BoW approaches on examples taken from the test sets of two datasets. Predictions in black color in non-bold/non-italic font were a part of the ground truth. Predictions in bold italics were such that their co-occurrence with other ground truth labels in the test set was novel, i.e. those co-occurences were never seen in the training set. Predictions in light gray color were not a part of the ground truth. DECAF offers much more precise recommendations on these examples as compared to other methods, for example AttentionXML, whose predictions on the last example are mostly irrelevant, e.g. focusing on labels such as "Early United States commemorative coins", instead of those related to the New Zealand dollar. DECAF is able to predict labels that never co-occur in the training set due to its inclusion of label text in to the classifier. For example, in the last example, the label "Australian dollar" never occurred with the other ground truth labels in the training set i.e. had no common training instances with the rest of the ground truth labels. Similarly, in the first example, the label "Panzer Dragoon Orta" never occurred together with other ground truth labels yet DECAF predicted these labels correctly while the other XML algorithms could not do so.

Algorithm	**Predictions**
	LF-AmazonTitles-131K
Document	**Panzer Dragoon Zwei**
DECAF	Panzer Dragoon, Action Replay Plus, Sega Saturn System - Video Game Console, The Legend of Dragoon , ***Panzer Dragoon Orta***
Astec	Guns of the Wehrmacht 1933-1945 (2006), Mission Barbarossa, Stug III & IV-Assault Guns, Tiger: Heavy Tank Panzer VI, Blitzkrieg
Bonsai	Playstation 1 Memory Card (1 MB), Mission Barbarossa, PlayStation 2 Memory Card (8MB), The Legend of Dragoon, Blitzkrieg
MACH	Mission Barbarossa, German Military Vehicles, Guns of the Wehrmacht 1933-1945 (2006), Stug III & IV - Assault Guns, Panther - The Panzer V (2006)
AttentionXML	Panther - The Panzer V (2006), Mission Barbarossa, German Military Vehicles, Stug III & IV - Assault Guns, The Legend of Zelda: A Link to the Past
Slice	Tiger: Heavy Tank Panzer VI, Stug III & IV - Assault Guns, Guns of the Wehrmacht 1933-1945 (2006), German Military Vehicles, The Legend of Dragoon
Document	**Wagner - Die Walkure / Gambill, Denoke, Rootering, Behle, Jun, Vaughn, Zagrosek, Stuttgart Opera**
DECAF	Wagner - Siegfried / West, Gasteen, Göhring, Schöne, Waag, Jun, Herrera, Zagrosek, Stuttgart Opera, Wagner - Gotterdammerung / Bonnema, DeVol, Iturralde, Kapellmann, Bracht, Westbroek, Zagrosek, Stuttgart Opera, Wagner - Der Fliegende Holländer (1986), ***Wagner - Gotterdammerung / Treleaven, Polaski, Salminen, Struckmann, Matos, von Kannen, de Billy, Barcelona Opera (2005)***, Seligpreisung
Astec	Lehar: Die lustige Witwe (2004) , Wagner - Der Fliegende Holländer (1986), Wagner: Der Ring des Nibelungen - Complete Ring Cycle (Levine, Metropolitan Opera) (2002), Verdi: Macbeth (The Metropolitan Opera HD Live Series) (2008), Richard Wagner - Der fliegende Holländer (2008)
Bonsai	Puccini: Tosca (Royal Opera House 2011) (2012), Weisser Holunder, Wagner - Die Meistersinger von Nurnberg / Heppner, Mattila, Morris, Pape, Allen, Polenzani, Levine, Metropolitan Opera, Wagner: Parsifal / Metropolitan Opera (1993), Un Giorno Di Regno (2010)
MACH	Puccini: Tosca (Royal Opera House 2011) (2012), Puccini: La Boheme, The Main Event [ORIGINAL RECORDINGS REMASTERED] 2CD SET, Super Mario World , Wagner - Gotterdammerung / Bonnema, DeVol, Iturralde, Kapellmann, Bracht, Westbroek, Zagrosek, Stuttgart Opera
Parabel	Wagner - Der Fliegende Holländer (1986), Die Meistersinger Von Nurnberg [Blu-ray] (2011), Un Giorno Di Regno (2010), Puccini: La Boheme, Wagner - Die Meistersinger von Nurnberg / Heppner, Mattila, Morris, Pape, Allen, Polenzani, Levine, Metropolitan Opera
AttentionXML	Puccini: La Boheme, Verdi: Macbeth, Puccini: Tosca (Royal Opera House 2011) (2012), Tannhauser (2008) , Wagner - Gotterdammerung / Treleaven, Polaski, Salminen, Struckmann, Matos, von Kannen, de Billy, Barcelona Opera (2005)

Slice	Rossini - Semiramide / Conlon, Anderson, Horne, Metropolitan Opera (1991), Richard Wagner - Der fliegende Holländer (2008) , Wagner - Der Fliegende Holländer (1986), Wagner - Gotterdammerung / Jones, Mazura, Jung, Hubner, Becht, Altmeyer, Killebrew, Boulez, Bayreuth Opera (Boulez Ring Cycle Part 4) (2005), Rossini - Il Turco in Italia / Bartoli, Raimondi, Macias, Rumetz, Schmid, Welser-Most, Zurich Opera (2004)
LF-WikiSeeAlsoTitles-320K	
Document	**New Zealand dollar**
DECAF	***Economy of New Zealand***, Cook Islands dollar, Politics of New Zealand , Pitcairn Islands dollar, ***Australian dollar***
Astec	Coins of the Australian dollar, List of banks in New Zealand, Constitution of New Zealand, Independence of New Zealand, History of New Zealand
Bonsai	Military history of New Zealand, History of New Zealand, Timeline of New Zealand history, Timeline of the New Zealand environment, List of years in New Zealand
MACH	Military history of New Zealand, History of New Zealand, List of years in New Zealand, Timeline of the New Zealand environment, Timeline of New Zealand's links with Antarctica
Parabel	Early United States commemorative coins, Environment of New Zealand, Half dollar (United States coin), History of New Zealand, Conservation in New Zealand
AttentionXML	Coins of the Australian dollar, Early United States commemorative coins, Half dollar (United States coin), Agriculture in New Zealand, Politics of New Zealand
Slice	Timeline of New Zealand's links with Antarctica, Coins of the Australian dollar, Early United States commemorative coins, List of New Zealand state highways, Timeline of the New Zealand environment