Interacting Knowledge Sources, Inspection and Analysis: Case-studies on Biomedical Text Processing

Parsa Bagherzadeh and Sabine Bergler
CLaC Labs, Concordia Univeristy
Montréal, Canada
{p_bagher, bergler}@cse.concordia.ca

Abstract

In this paper we investigate the recently proposed multi-input RIM for inspectability. This framework follows an encapsulation paradigm, where external knowledge sources are encoded as largely independent modules, enabling transparency for model inspection.

1 Introduction

Deep learning with pre-trained language models is widely used in a variety of NLP tasks. Such models are often trained using an end-to-end approach, which assumes that the structure of the network is sufficient to capture the inductive bias from adequate data, using a gradient-descent approach.

Deploying language models for applications such as biomedical text processing, however, poses two main challenges. First, adequate training data is often not available due to the cost of annotation by domain experts and, indeed, the exploratory nature of most biomedical tasks. Second, the biomedical domain uses many terms with a more specialized meaning and language models often cannot demonstrate an understanding of this meaning. For instance, consider Example 1:

(1) *TNFAIP3 is a gene whose expression is induced by the tumor necrosis factor.*

Example 1 defines TNFAIP3 as a *gene* in a copula construction and its *expression* in a relative clause. We mask the token *gene* and ask BioBERT (Lee et al., 2020) (as a masked language model) to predict the masked token. The top 5 predictions are *simulation, coordinates, diffusion, heat,* and *pyramid*. The default solution is to continue the pre-training process of the language models on larger corpora, which is often prohibitive for users with limited access to computational resources. Moreover, biomedical terminologies are always growing and enough textual data might not be available for new concepts. A cheaper option is to use extant quality knowledge sources, in form of ontologies, hierarchies, or specialized gazetteer lists. In the developing field of knowledge injected models, these high-quality, general knowledge sources are expected to work in tandem to address the problem at hand (Søgaard and Goldberg, 2016; Levy and Goldberg, 2014; Tian et al., 2020; Bagherzadeh and Bergler, 2020).

Current knowledge injection models introduce specialized layers to language models or add new training objectives. For instance, (Peters et al., 2019) developed KnowBERT (an extension to BERT), to inject information from WordNet into an intermediate layer of BERT, which requires retraining the BERT model. To leverage medical concepts from the medical knowledge base ULMS, (Hao et al., 2020) added an auxiliary prediction task to the BERT model by training a binary classifier to predict relations between two concepts.

The chosen knowledge source is an integral part of these language models making their contribution difficult to inspect. In precision-oriented applications, users require to understand why and how a prediction is made. Inspection of a system helps developers to detect fallacies in a system and provide insight for the improvement of the system (Verma et al., 2020; Amini and Kosseim, 2019) and users to trust the autonomous system.

Different knowledge sources might include overlapping, contradicting, or differently aimed knowledge,[1] and the wrong components may hamper rather than enhance the learning progress (Glasmachers, 2017). In contrast, if the knowledge sources can adapt to the domain separately and only interact sparingly, the resulting model is expected to be robust (Schölkopf et al., 2012; Goyal et al., 2019)

(Bagherzadeh and Bergler, 2021b) introduced

[1] A *size* feature may encode *height, width,* or *volume*, for instance.

Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 447–456

the multi-input Recurrent Independent Mechanisms (mi-RIM), which comprise a set of independent and competitive recurrent modules that operate individually on different knowledge embeddings. They report that adding gazetteer and POS tag modules consistently improves performance on a variety of tasks. The mi-RIM architecture injects knowledge in form of different knowledge sources as largely independent modules. They suggest that activation visualizations of each module allow for inspection but do not further probe whether they provide enough insight to explain the system predictions.

This paper offers an in-depth analysis of the contribution of several knowledge sources in several tasks within the mi-RIM framework. We confirm that once modules compete, they specialize to focus on different parts of the input and inspect the competition patterns during processing as a source for explanation and user feed-back.

2 Overview of mi-RIM

Recurrent independent mechanisms (RIM), first introduced by (Goyal et al., 2019), is a modular architecture that models a dynamic system by dividing it into M recurrent units, all of which operate on the same input sequence. The units are selective, i.e they chose to use or ignore their input, and are able to communicate with one another. The standard RIM model was adapted to the language domain and extended by (Bagherzadeh and Bergler, 2021b) to multi-input RIMs (mi-RIM) to allow different modules to operate on different inputs. Since we use mi-RIMs in our case study, a brief overview of their model is provided in the following.

Input selection Each module R_m augments the token input x_t^m to $X_t^m = x_t^m \oplus \mathbf{0}$, where $\mathbf{0}$ is an all-zero vector and $\oplus$ denotes row-level concatenation. Then, using an attention mechanism, unit R_m selects input:

$$A_t^m = softmax\left(\frac{h_{t-1}^m W_m^{query}(K_m)^T}{\sqrt{d_h}}\right)V_m \quad (1)$$

where $h_{t-1}^m W_m^{query}$ is the *query*, $K_m = X_t^m W_m^{key}$ is the *key*, and $V_m = X_t W_m^{val}$ is the *value* in the attention mechanism (Vaswani et al., 2017). If the input x_t is considered relevant to the task, the attention mechanism in Equation 1 assigns more weight to it (selects it), otherwise more weight will be assigned to the null input. The $softmax$ values of Equation 1 determine a ranking for the modules and a subset S_t of the k highest ranked units. Among M units, those with the least attention on the null input are the active units. The selected input A_t^m determines a temporary hidden state $\tilde{h}_t^m$ for the active units:

$$\tilde{h}_t^m = R_m(h_{t-1}^m, A_t^m) \quad m \in S_t \quad (2)$$

where $R_m(h_{t-1}^m, A_t^m)$ denotes one iteration of updating the recurrent unit R_m based on previous state h_{t-1}^m and current input A_t^m. The hidden states of the inactive units R_m ($m \notin S_t$) remain unchanged:

$$h_t^m = h_{t-1}^m \quad m \notin S_t \quad (3)$$

Communication To obtain the actual hidden states h_t^m, the active units communicate using an attention mechanism:

$$h_t^m = softmax\left(\frac{Q_{t,m}(K_{t,:})^T}{\sqrt{d_h}}\right)V_{t,:} + \tilde{h}_t^m \quad m \in S_t \quad (4)$$

where

$$Q_{t,m} = \tilde{h}_t^m \tilde{W}_m^{query}$$

$K_{t,:}$ is the row-level concatenation of all $K_{t,m}$ ($m = 1, \ldots, M$) defined as:

$$K_{t,m} = \tilde{h}_t^m \tilde{W}_m^{key}$$

and $V_{t,:}$ is the row-level concatenation of all $V_{t,m}$ ($m = 1, \ldots, M$) defined as:

$$V_{t,m} = \tilde{h}_t^m \tilde{W}_m^{val}$$

Both the key $K_{t,:}$ and the value $V_{t,:}$ depend on the temporary hidden states of all units, therefore h_t^m in Equation 4 is determined by attending to all units.

As pointed out by (Wiegreffe and Pinter, 2019), the attention-based models can provide explanation when the attention is applied to individual elements in an input sequence, rather than contextualized ones. Similarly, input selection attention for the mi-RIM architecture also operates on raw input encoding. When an input is considered to be relevant to the task, the corresponding module is allowed to be active and to be updated with its additional, specialized input. The activation patterns for each module are traced to inspect what elements in the input sequence were used.

3 Tasks

This paper focuses on the biomedical domain for two reasons. First, effective knowledge components for many biomedical tasks are well-defined and high quality knowledge sources are freely available. Second, model inspection is essential in the biomedical domain, particularly for health related applications. The following four tasks are considered here:

BC7-3 or BioCreative VII Task 3 is a sequence labelling task for detecting spans of drug mentions in tweets.[2] Example 2 shows a tweet with one mention of a drug (*Zantac*), and Example 3 is a tweet with no drug mention. BC7-3 is evaluated by F1 score.

(2) *Sprite and Zantac. was not the best idea.*

(3) *The meds have arrived!!*

BC2-1a or BioCreative II Task 1a concerns extraction of gene mentions in biomedical texts (Smith et al., 2008). Example 4 contains a gene mention (*alpha fetoprotein*) and Example 5 does not contain any gene mentions. The performance is evaluated by F1 score.

(4) *False positive amniotic fluid alpha fetoprotein levels resulting from contamination with fetal blood*

(5) *Teratological study of etoperidone in the rat and rabbit.*

SM21-1a concerns adverse drug event classification of tweets (Klein et al., 2021). Examples 6 and 7 are instances of tweets with and without an adverse drug event respectively. The task is evaluate by F1 score.

(6) *worst was the psychotic episode when starting #paxil*

(7) *need nicotine*

SM21-6 is a three-way classification task of tweets containing CoVID symptom mentions (Klein et al., 2021). In *self-report* tweets (Example 8), posters mention a Covid symptom that they have experienced. *Nonpersonal-reports* however report a symptom for someone else (Example 9), and *News-mentions* are symptom reports that are intended to raise awareness (Example 10). This task is evaluated by the micro-F1 score (μF1) of the *self-reports* and *Nonpersonal-reports* classes.

(8) *I am still breathless upon exertion, but my shortness of breath and cough that used to happen all the time*

(9) my mom was denied a coronavirus test even tho she has a high fever and corona symptoms.

(10) *Many of these patients with post-#COVID19 fatigue will have an abnormal tilt table test and a form of #dysautonomia.*

Note that since the gold labels for BC7-3, SM21-1a, and SM21-6 have not been disclosed, we report the result on a hold out set (20% of the training data).

4 Knowledge sources

For reproducibility, we use off the shelf knowledge sources that are available from standard NLP and deep learning environments, or can be easily accessed from online ontological repositories.

Morphology Drug or gene names often have a specific morphology, favoring certain prefixes, suffixes, etc. The suffix *-statin*, for instance, is observed in drug names of this type, including *torvastatin, lovastatin,* and *pravastatin*. Such morphological information[3] can be used to detect possible mentions of drugs or genes that have not been observed in training data, or those that are not present in drug or gene lists.

Word embeddings BioBERT (Lee et al., 2020) word embeddings provide a meaning representation based on co-occurrence statistics.

POS Part-of-speech tags are the most widely used linguistic feature and are available from many standard NLP environments. POS tags provide useful information such as types of pronouns and tense for verbs, allimportant clues for sequence labeling and text classification.

[2] https://biocreative.bioinformatics.udel.edu/tasks/biocreative-vii/track-3/

[3] Note that the subtoken input for BERT models can detect these regularities.

Drug DrugBank includes commercial drug names as well as the scientific names of their active ingredients (Wishart et al., 2018).[4]

Anatomy Body part mentions are important evidence for detection of health experience mentions, in particular adverse drug reactions or disease symptoms. Relevant anatomy terms are extracted from sub-tree A of MeSH (Lipscomb, 2000) into a gazetteer list.

Disease Disease mentions are important evidence for adverse drug event classification and self-reports of disease symptoms. A gazetteer was compiled from subtree C in MeSH which includes terms for *infections, wounds, injuries, pain, etc.*.

Gene To identify mentions of genes, we compile a gazetteer list from the NCBI Gene database. [5]

5 Implementation

Preprocessing We preprocess the data using a GATE pipeline (Cunningham et al., 2002) and use the ANNIE tweet tokenizer as well as the hashtag tokenizer for BC7-3, SM21-6, and SM21-1a data, and the PennBio tokenzier for BC2-1a data. For tweet tasks, we remove URLs and user mentions.

Modules Each module m comprises an embedder E_m (for encoding a knowledge source) and a recurrent unit R_m. For the token at position t, E_m emits its knowledge representation $x_t^m \in \mathbb{R}^{d_{in}^m}$ and $X_t^m = x_t^m \oplus \mathbf{0}$ is used as input to the recurrent layer R_m.

Morph. Following (Kim et al., 2016) we use a character-level Convolutional Neural Network to obtain a morphological representation for each token. We use multiple convolution filters $\mathcal{F}_l$ with different lengths of $l \in \{2, 3, 4, 5\}$. The resulting character-based representations are in a 100-dimensional space.

Word emb. We use the last layer of BioBERT (Lee et al., 2020) to obtain word embeddings. The embeddings are then used as input to a LSTM.

POS Following (Bagherzadeh and Bergler, 2021a), we pre-train POS tags using Word2Vec (Mikolov et al., 2013) to initialize an embedding layer.

Gazetteer modules Each gazetteer list is embedded by two trainable vectors, one corresponding to a match against the gazetteer, and another one for a mismatch. The gazetteer embedings are then used as input to simple RNN units rather than LSTMs.

All NN components are implemented using PyTorch (Paszke et al., 2017), using cross-entropy to calculate loss and the Adam optimizer (Kingma and Ba, 2015).

6 Experiments

We report three sets of experiments. First, we inject different knowledge sources using the mi-RIM model, and evaluate their effectiveness in terms of precision (P), recall (R), and F1 scores. Second, we inspect the activity profiles of the modules for several samples. Third, we provide a quantitative analysis on whether module activity can be used to provide insight into model's predictions using Total Variation Distance (TVD) (Jain and Wallace, 2019) as a measure of change between output distributions.

6.1 Performance

We investigate whether each knowledge source can contribute to each of the different tasks and how different knowledge sources cooperate or whether redundancy or contradicting values decrease performance.

Carefully selected sources Table 1 reports performance when using the four modules *Word Emb., Morph, Drug,* and *POS* on the drug mention detection task BC7-3. Three different competition settings are reported, $k \in \{1, 2, 3\}$. Greater k values are not reported, because performance drops.

Modules	k	P	R	F1
Word Emb.	1	.60	.69	.64
System BC7-3:	3	.67	.68	.67
Word Emb., Morph, Drug, POS	2	.69	**.70**	**.70**
	1	**.70**	.63	.66

Table 1: 4 module mi-RIM system on BC7-3

Table 2 reports performance when using the four modules *Word Emb., Morph, Gene,* and *POS* on the gene mention detection task BC2-1a. Both

[4] https://go.drugbank.com/drugs/DB00863 identifies *Zantac* (Example 2) as a drug mention

[5] *alpha fetoprotein* in Example 4 is identified by https://www.ncbi.nlm.nih.gov/gene/174

BioCreative mention detection tasks show best performance when only two modules can be active at a time.

Modules	k	P	R	F1
Word Emb.	1	.85	.84	.84
System BC2-1a:	3	.86	.86	.86
Word Emb., Morph, Gene, POS	2	**.89**	.85	**.87**
	1	.82	**.89**	.85
SOTA (Lee et al., 2020)	–	.85	.83	.84

Table 2: 4 module mi-RIM system on BC2-1a

Table 3 shows a similar behaviour for five modules on the adverse drug reaction classification task SM21-1a.

Modules	k	P	R	F1
Word Emb.	1	.56	.63	.58
System SM21-1a:	3	.57	.67	.62
Word Emb., Drug,	2	.59	.70	**.64**
Disease, Anatomy, POS	1	**.53**	.74	.63

Table 3: 5 module mi-RIM system on SM21-1a

Modules	k	μP	μR	μF1
Word Emb.	1	92	.93	.92
System SM21-6:	3	.94	.95	.94
Word Emb., Disease,	2	.95	.95	**.95**
Anatomy, POS	1	.92	**.96**	.94

Table 4: 4 module mi-RIM system on SM21-6

The external knowledge sources consistently improve performance across the tasks. Peak performance (F1) is reached for $k = 2$.

This forms a proof of concept that external knowledge sources can be harnessed in fixed configurations tailored to specific tasks. This has been shown in different architectures as well.

Some redundant sources We claim that the mi-RIM architecture can combine redundant and possibly contradicting knowledge sources with no ill effects and offer as justification a seven module system combining all the knowledge sources introduced here and comparing its performance on all the tasks.

Table 5 shows that in fact, the overabundance of these resources slightly lowers performance sometimes (SM21-6), but sometimes improves performance further. We consider this a strong endorsement of the claim.

6.2 Visualization of activation patterns

When $k < M$, only a subset of modules is allowed to contribute and the modules enter into competition mode. As claimed by (Goyal et al., 2019), competition for activity makes the modules to focus on specific parts of the input. We probe activation patterns for varying k.

	BC7-3			*BC2-1a*			*SM21-1a*			*SM21-6*		
k	P	R	F1	P	R	F1	P	R	F1	P	R	μF1
3	.69	.68	.68	.87	.86	.86	.59	.66	.63	.93	.93	.93
2	.70	.70	.70	**.88**	**.87**	**.87**	**.62**	.69	**.66**	.94	**.95**	.94
1	.69	.68	.67	.83	.88	.85	.55	.73	.64	.91	.94	.93

Table 5: Seven module system (Word Emb., Morph, POS, Drug, Disease, Anatomy, Gene) on all tasks

Figures 1–9 show the activation patterns for the four tasks. In each figure's caption, the input is given and the gold spans are indicated as underlined text. Moreover, the correctly tagged tokens are indicated by solid boxes, and the false positives by dashed boxes.

Sequence labeling tasks Figure 1 is an instance of a true-positive classification for BC7-3 (drug mention span detection). When $k = 3$, the *Drug* module is only active when there is a match with the Drug gazetteer; *Fentanyl*. Other modules however are always active which means that they are not focusing on a specific part of the input. On the other hand, when $k = 2$ the *Morph* module is active only for the drug mention, which means that the competition has made the module more focused. For this example, activation patterns of the seven module system for $k = 2$ are identical. Note that for $k = 1$, only the *Drug* module is active for the mention of Fentanyl, indicating that the model prioritizes the high quality drug gazetteer over the general *Morph* module when a choice has to be made.

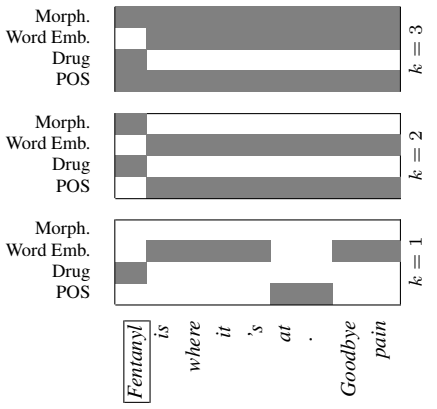

Figure 1: True positive case for BC7-3: *Fentanyl is where it's at. Goodbye pain*

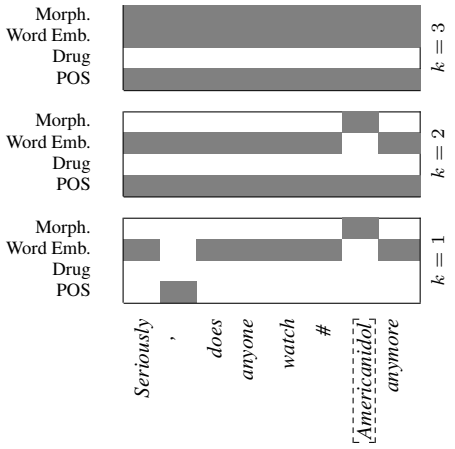

Figure 2: False positive case for BC7-3: *Seriously, does anyone watch #Americanidol anymore*

Figure 2, on the other hand, shows a false-positive case for BC7-3. *Americanidol* has been identified as a drug mention. When $k < 3$, the *Morph* module is active only for the supposed drug mention, suggesting the cause of the misclassification: the token *Americanidol* is a good candidate for a drug name based on its morphological structure: it is capitalized, a composite name, and ends in *ol*. Inspection of such cases using the activation patterns provides valuable insight for users and developers, and allows users to give nuanced feedback to developers capitalizing on their expertise. For instance, expert users might suggest enhancing the tokenizer to split hashtags, for instance, *#Americanidol* into [*#, American, idol*] instead of [*#, Americanidol*]. For this example, activation patterns of the seven module system for $k = 2$ are identical.

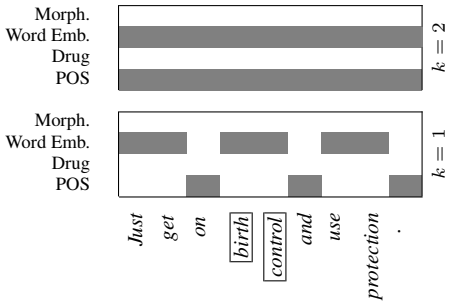

Figure 3: True positive case for BC7-3: *Just get on birth control and use protection*

Figure 3 shows another example of a correctly tagged drug. The term *birth control* is not recognized by the Drug gazetteer, therefore, the *Drug* module is never active. In competition mode ($k = 1$ or $k = 2$), the *Morph* module is also never active, however, the *Word Emb.* module at $k = 1$ is active for the span of *birth control*, and contributes its input.

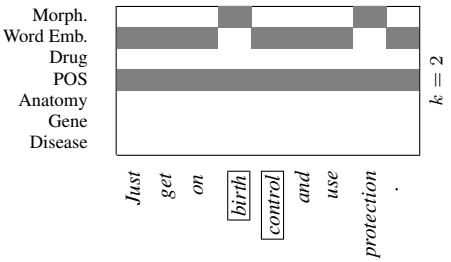

Figure 4: True positive case for BC7-3, 7-modules system: *Just get on birth control and use protection*

This example demonstrates the advantage of redundant capability in a highly competitive context ($k = 1$) and showcases the overgeneralization process for the deep modules *Morph* and *Word Emb.* in the context of redundancy. If a knowledge source (in this case a gazetteer list) has limited expertise, other modules can compensate (capability) but a danger of synchronization exists for fully redundant modes. Figure 4 demonstrates that an abundance of modules can lead to more specialized patterns.

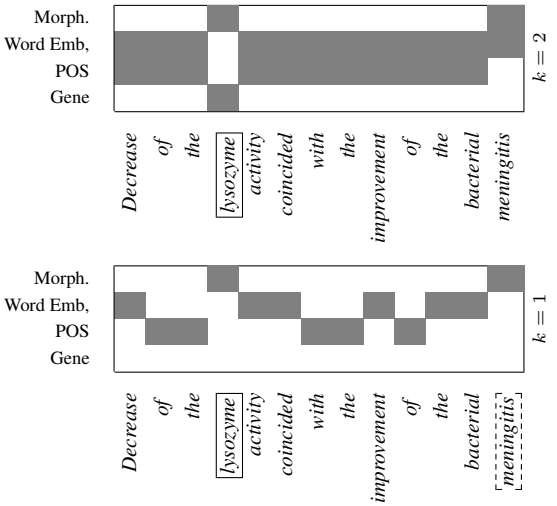

Figure 5: True positive and false positive cases for BC2-1a: *Decrease of the lysozyme activity coincided with the improvement of the bacterial meningitis.*

Figure 5 shows the activation patterns for a sample from BC2-1a. We see a behaviour similar to those for the drug span task; once modules are in competition, they are more focused on specific parts of the input. An interesting observation here

is the difference between predictions when k has different values. When $k = 2$ and $k = 3$, the model correctly tags only *lysozyme* as a gene mention. However, when $k = 1$, the model makes a false positive prediction for *meningitis* as well. Note that when $k = 1$, the *Morph* module is active for the token *meningitis*. In this case, the *Morph* module has won the competition and has forced the *Word Emb.* module to be inactive, which is the result of intense competition between modules. This however is not the case when $k = 2$ and both *Morph* and *Word Emb.* modules are active and contribute to a correct prediction.

Document-level tasks Figure 6 shows a true-positive case from SM21-1a. Similar to previous patterns, the more the modules are in competition (smaller k), the more their activities are sparse and focused. The *Anatomy* module is active for the body parts mentions *tendons* and *hands*, and the *Disease* module is active for *pain* and *ruptured*. Note that the *Word Emb.* module is inactive for tokens that are matched with gazetteer lists. On the other hand, the *Word Emb.* module is active for phrases/tokens such as *barely usable* and *damned*, as other evidence for the adverse event classification. The seven modules system again leads to a more specialized pattern in Figure 7.

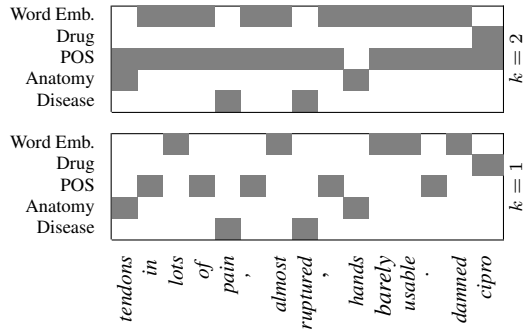

Figure 6: True positive case for SM21-1a: *tendons in lots of pain, almost ruptured & hands barely usable. damned cipro*

Figure 8 on the other hand shows a false-positive case for SM21-1a (adverse drug effect mentions). Note that the tweet is an example of a text with figurative language. When $k = 1$, the *Word Emb.* module is active for the phrases *the marbles made my* and *look pink*. It seems the activity on *look pink* has mislead the model to make a false-positive prediction.

The seven module system shows unfortunately

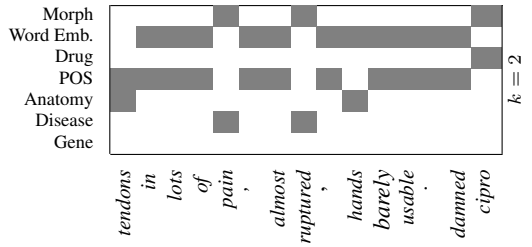

Figure 7: True positive case for SM21-1a, 7 module system: *tendons in lots of pain, almost ruptured & hands barely usable. damned cipro*

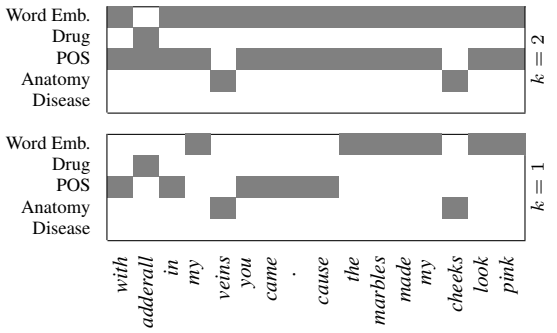

Figure 8: False positive case for SM21-1a: *with adderall in my veins you came. cause the marbles made my cheecks look pink.*

that the abundance of modules is not sufficient to rectify the error, even though the module activations change in certain parts: *Word Emb.* relieves *POS* for *adderall* and *Morph* relieves *Word Emb.* for *marbles*.

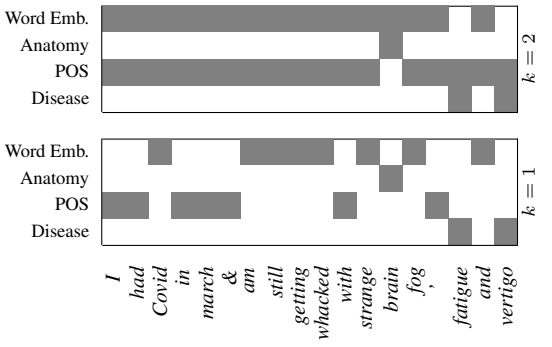

Figure 9: True positive case for SM21-6: *I had Covid in march & am still getting whacked with strange brain fog, fatigue and vertigo*

Finally, an example of a true-positive prediction for SM21-6 (self-reports of CoVID symptoms) is provided in Figure 9. There are three symptom mentions such as *brain fog*, *fatigue*, and *vertigo*, and the *Disease* module is active for the latter two. Interestingly, when $k = 1$, the *Word Emb.* mod-

ule is active for the token *Covid* and has won the competition against the *Disease* module. Further analysis of the activation patterns revealed that at 77% of the time, the *Disease* module is active only when a token corresponds to a symptom. However, when the actual name of the disease is mentioned, the *Word Emb.* module is active. We surmise, due to the frequent occurrence of terms such as *Covid* and *Corona* in the training set, the *Word Emb.* module has learned to focus on such tokens, and the *Disease* module covers the more lexically-diverse symptom mentions. More module choice in the seven modules system for $k = 2$ leads to three changes: *Morph* is active for *Covid*, *whacked*, and *fatigue*, relieving *Word Emb.* at these time points.

As argued by (Wiegreffe and Pinter, 2019) and (Serrano and Smith, 2019) the attention mechanism fails to provide explanation, since it operates on the contextualized representations. This, however is not the case for the models in the mi-RIM architecture since module activity is the result of the input selection mechanism, assuring that when a module is not active at a particular time-step, it is guaranteed that its input is not used.

6.3 Quantitative analysis of module activity

(Jain and Wallace, 2019) argue that if attention weights provide insight into why a prediction is made, removing a token accorded with a high attention weight has to significantly change the output distribution. In other words, the question is *had we not attended to an element, would we have the same output?* A yes to this question means there is no association between the input with high attention weight and the prediction. A no to this question however shows that the input is crucial for making that prediction. Thus (Jain and Wallace, 2019) perform *leave-one-out* experiments and calculate the Total Variation Distance (TVD) as a measure of change between output distributions, which is defined as:

$$\text{TVD}(\hat{y}(\mathbf{X}), \hat{y}(\mathbf{X}_{-t})) = \frac{1}{2}\sum_{i=1}^{|\mathcal{K}|} |\hat{y}(\mathbf{X})_i - \hat{y}(\mathbf{X}_{-t})_i| \quad (5)$$

where $|\mathcal{K}|$ is the number of classes, $\hat{y}(\mathbf{X})$ is the output distribution when all tokens are in the input, and $\hat{y}(\mathbf{X}_{-t})$ is the output distribution when token at position t is removed from the input. As an example, suppose the initial output distribution for a 3-class problem is:

$$\hat{y}(\mathbf{X})_1 = 0.7\ ,\ \hat{y}(\mathbf{X})_2 = 0.1\ ,\ \hat{y}(\mathbf{X})_3 = 0.2$$

and the secondary distribution is:

$$\hat{y}(\mathbf{X}_{-t})_1 = 0.1\ ,\ \hat{y}(\mathbf{X}_{-t})_2 = 0.8\ ,\ \hat{y}(\mathbf{X}_{-t})_3 = 0.1$$

then, the TVD value is 0.7 which is a significant variation in the distribution of the output.

When modules are in competition, they are only active for specific parts of the input that are relevant to the task, and this activity pattern can be inspected to provide some explanation for behavior of the model. To support our claim, we use a similar set of *leave-one-out* experiments as (Jain and Wallace, 2019). Suppose a module is active at time-step t. We manually deactivate the module only for this time-step, and then measure the TVD value. As an example, consider Figure 6, when $k = 2$. The module *Disease* is active for tokens *pain* and *ruptured*. We deactivate the module only at those two time-steps (the rest of the modules keep their activity state) as two *leave-one-out* cases. Note that at time-step t, the inactive modules simply retain their previous hidden states (see Equation 3).

As a consequence, performance of the model is expected to be affected due to changes in output distribution. The performance change can also be used as a measure of the importance of module activity. If the activity of a module is important towards a prediction, deactivating it has to change the performance. When performing leave-one-out experiments, we report these changes by ΔF1. Table 6-8 report the average TVD values,[6] as well as corresponding ΔF1 values.

		Morph		*Word Emb.*		*Drug*		*POS*	
		TVD	ΔF1	TVD	ΔF1	TVD	ΔF1	TVD	ΔF1
k	3	.12	-.02	.08	-.02	.14	-.03	.01	.00
	2	.36	-.16	.16	-.06	.30	-.21	.08	-.01
	1	.42	-.23	.20	-.11	.54	-.32	.14	-.03

Table 6: Mean TVD and ΔF1 on BC7-3

We can assess the overall value of a module for a task by the greatest difference in TVD or ΔF1 values. In Table 6, *Drug* outranks *Morph* followed by *Word Emb.* and *POS*. In fact, *POS* appears to be a marginal module of no great impact on the analyzed examples.

[6] averaged over all leave-one-out experiments

		Morph.		*Word Emb.*		*Gene*		*POS*	
		$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1
	3	.18	-.05	.10	-.02	.12	-.02	.06	.00
k	2	.48	-.23	.22	-.09	.26	-.13	.12	-.04
	1	.62	-.41	.32	-.18	.54	-33	.16	-.05

Table 7: Mean TVD and ΔF1 on BC2-1a

For Table 7, *Morph* outranks *Drug*, outlining that the results are specific to the tasks and data.

For document-level tasks, changes in the output distribution observed are smaller due to the fact that users often report several symptoms for Covid, and rejecting one of the symptoms as input to the modules does not prevent the model from correct classification, since other evidence still exist. Still, we note that Table 8 for SM21-6 (self-reports of Covid symptoms) shows that for this task *Anatomy* outranks *Word Emb.* followed by *Disease* and finally *POS*, but that the differences are very small.

		Word Emb.		*Disease*		*Anatomy*		*POS*	
		$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1
	3	.08	-.01	.04	.00	.06	.00	.02	.00
k	2	.12	-.02	.10	-.02	.14	-.03	.08	-.01
	1	.18	-.05	.20	-.07	-.28	-.12	.18	-.05

Table 8: Mean TVD and ΔF1 on SM21-6

The different behaviour of the *Anatomy* module on SM21-a (adverse drug effects) is interesting; deactivation has a greater effect since mentions of body parts are very frequent in reports of adverse drug effects.

		Word Emb.		*Drug*		*Disease*		*Anatomy*		*POS*	
		$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1	$\overline{\text{TVD}}$	ΔF1
	3	.08	-.02	.06	-.01	.06	-.01	.06	.00	.00	.00
k	2	.12	-.03	.18	-.04	.10	-.03	.09	-.02	.04	.00
	1	.22	-.11	.42	-.24	.22	-.10	.16	-.05	.06	-.01

Table 9: Mean TVD and ΔF1 on SM21-1a

7 Conclusion

We have demonstrated the potential of modules for visualizations for inspection: mean TVD values and $\Delta F1$ provide insight into general module importance for tasks, module activation patterns provide some insight into system behavior for individual samples, such as error cases. With the seven module system, we demonstrate that ineffective knowledge sources can be combined without harm in the mi-RIM architecture.

The experiments suggest that the encapsulation approach for leveraging knowledge sources in the mi-RIM architecture is promising but still has to be applied to more modules to gauge complexity implications.

Acknowledgements

We would like to thank Hessam Amini for insightful discussions and comments on explainability and model interpretation.

References

Hessam Amini and Leila Kosseim. 2019. Towards explainability in using deep learning for the detection of anorexia in social media. In *Natural Language Processing and Information Systems*, pages 225–235. Springer International Publishing.

Parsa Bagherzadeh and Sabine Bergler. 2020. CLaC at SMM4H 2020: Birth defect mention detection. In *Proceedings of the Fifth Social Media Mining for Health Applications Workshop & Shared Task*.

Parsa Bagherzadeh and Sabine Bergler. 2021a. Leveraging knowledge sources for detecting self-reports of particular health issues on social media. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*.

Parsa Bagherzadeh and Sabine Bergler. 2021b. Multi-input Recurrent Independent Mechanisms for leveraging knowledge sources: Case studies on sentiment analysis and health text mining. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 108–118.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Tobias Glasmachers. 2017. Limits of end-to-end learning. In *Asian Conference on Machine Learning*, pages 17–32.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. 2019. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*.

Boran Hao, Henghui Zhu, and Ioannis Paschalidis. 2020. Enhancing clinical BERT embedding using a biomedical knowledge base. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 657–661.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M Rush. 2016. Character-aware neural language models. In *Thirtieth AAAI conference on Artificial Intelligence*.

Diederik P Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR'15*.

Ari Z. Klein, Ivan Flores, Arjun Magge, Anne-Lyse Minard, Karen O'Connor, Abeed Sarker, Elena Tutubalina, Davy Weissenbacher, and Graciela Gonzalez-Hernandez. 2021. Overview of the Sixth Social Media Mining for Health applications (SMM4H) shared tasks at NAACL 2021. In *Proceedings of the Sixth Social Media Mining for Health Applications (SMM4H) Workshop & Shared Task*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308.

Carolyn E Lipscomb. 2000. Medical subject headings (MeSH). *Bulletin of the Medical Library Association*, 88(3).

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS 2017*.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. 2012. On causal and anticausal learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, pages 1255–1262.

Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951.

Larry Smith, Lorraine K Tanabe, Rie Johnson nee Ando, Cheng-Ju Kuo, I-Fang Chung, Chun-Nan Hsu, Yu-Shi Lin, Roman Klinger, Christoph M Friedrich, Kuzman Ganchev, et al. 2008. Overview of BioCreative II gene mention recognition. *Genome biology*, 9(2):1–19.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235.

Hao Tian, Can Gao, Xinyan Xiao, Hao Liu, Bolei He, Hua Wu, Haifeng Wang, and Feng Wu. 2020. SKEP: Sentiment knowledge enhanced pre-training for sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4067–4076.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Tejaswani Verma, Christoph Lingenfelder, and Dietrich Klakow. 2020. Defining explanation in an AI context. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 314–322.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20.

David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, et al. 2018. Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleic acids research*, 46(D1):D1074–D1082.