Enhancing Search Results with Semantic Annotation Using Augmented Browsing

Hong-Jie Dai[1,2*], **Wei-Chi Tsai**[3], **Richard Tzong-Han Tsai**[3*] **and Wen-Lian Hsu**[1,2*]
[1]Department of Computer Science, National Tsing-Hua University
[2]Intelligent Agent Systems Lab., Institute of Information Science, Academia Sinica
[3]Department of Computer Science and Engineering, Yuan Ze University
Republic of China (Taiwan)
hongjie@iis.sinica.edu.tw, {s986004@mail, thtsai@saturn}.yzu.edu.tw, hsu@iis.sinica.edu.tw

Abstract

In this paper, we describe how we integrated an artificial intelligence (AI) system into the PubMed search website using augmented browsing technology. Our system dynamically enriches the PubMed search results displayed in a user's browser with semantic annotation provided by several natural language processing (NLP) subsystems, including a sentence splitter, a part-of-speech tagger, a named entity recognizer, a section categorizer and a gene normalizer (GN). After our system is installed, the PubMed search results page is modified on the fly to categorize sections and provide additional information on gene and gene products identified by our NLP subsystems. In addition, GN involves three main steps: candidate ID matching, false positive filtering and disambiguation, which are highly dependent on each other. We propose a joint model using a Markov logic network (MLN) to model the dependencies found in GN. The experimental results show that our joint model outperforms a baseline system that executes the three steps separately. The developed system is available at https://sites.google.com/site/pubmedannotationtool4ijcai/home.

1 Introduction

The amount of biological literature is vast and growing rapidly. Large, well-curated biomedical resources, such as NCBI PubMed and EMBL-EBI, make available a massive volume of online articles for today's biologists to search through. In order to enhance readability of NCBI PubMed or EMBL-EBI search results, researchers have developed a range of web-based artificial intelligence (AI) systems which employ natural language processing (NLP) technology to identify and mark-up biomedical named entities, verbs and key relations among these components. For example, iHOP [Hoffmann and Valencia, 2005] and BIOSMILE [Dai *et al.*, 2008] provide enhancements to PubMed's retrieval of abstracts by organizing the results or highlighting specific entities in text. Unfortunately, compared to the popularity of PubMed or EMBL-EBI's online search interfaces, none the above services has been widely adopted by the biomedical research community. Part of the problem may be that all these advanced services have their own interfaces, some of which are very different from PubMed. Furthermore, the interfaces vary in terms of ease of use, and at the very least require users to navigate to an independent website, which may respond more slowly than PubMed search due to bandwidth and data access lag times.

An emerging approach, called augmented browsing[†], is being increasingly adopted on the Web and has been introduced into the life sciences. Built upon popular browser extension frameworks, this technology provides an effective means for dynamically adding supplementary information to a webpage without having users navigate away from the page. For example, ChemGM [Willighagen *et al.*, 2007] automatically recognizes chemical structure names found on a webpage and provides inline hypertext links to PubChem. Both ConceptWeb Linker [Mons *et al.*, 2008] and Reflect [Pafilis *et al.*, 2009] highlight several types of named entities and link them to ontologies on popular websites via popup windows. This type of service has a strong focus on ease of use and could possibly win over more converts in the biomedical research community.

In this paper, we describe our experience of constructing an AI system that augments PubMed's search page via a browser extension to provide users with enhanced customizable views. The system seamlessly integrates several biomedical text-mining subsystems' annotations with the PubMed search results. It can categorize sentences in an abstract into four sections (OBJECTIVE, METHOD, RESULT, and CONCLUSION), which provides an important feature for users to quickly recognize key biomedical information. For example, the key disease-related genes investigated in a paper tend to be mentioned in the results and conclusions sections. However, other parts of the abstract may contain mentions of disease-related genes that nonetheless are not actually experimented upon in the paper.

[*] Corresponding authors

[†] Augmented browsing refers to the experience of using a system that can automatically augment the information in webpages.

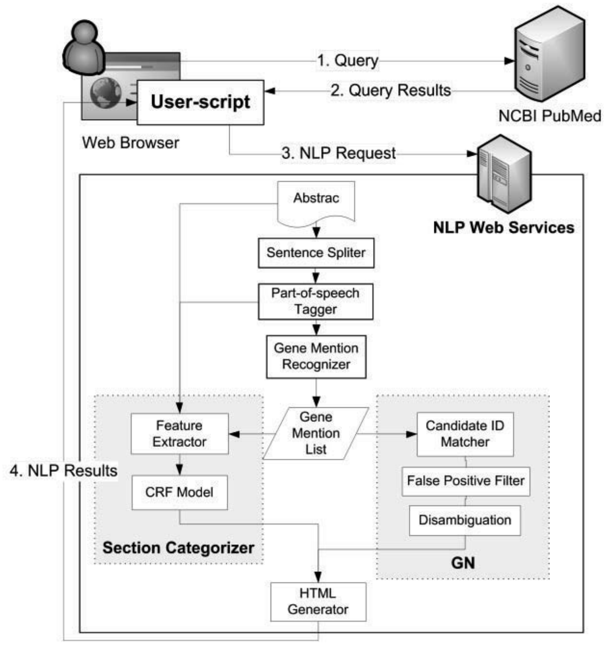


Figure 1. System flowchart.

Furthermore, our system enriches the original PubMed search results by highlighting disease and gene mentions and providing hyperlinks to MeSH/EntreGene database entries via entity normalization technology. By the normalization technology, we can then hyperlink entities in the abstract text to URLs on EntrezGene's website. When a user clicks on a hyperlinked entity, a pop-up window is displayed with corresponding detail information for the identified entity. A key advantage of using popups is that users can see information about an entity without having to navigate away from the current webpage. If needed, hyperlinks to more detailed information can be provided on the popup.

Gene normalization (GN) is the task of normalizing a textual gene mention to a unique gene database ID. GN is a complicated NLP process and usually includes four main steps: (1) gene mention recognition, (2) candidate ID matching, (3) false positive filtering and (4) disambiguation. Each step requires a different NLP subsystem. The entire GN procedure can be treated as a decision making system that considers each subsystem's output to determine the final decision. In addition to augmenting the PubMed search site with our embedded AI system, we present a novel approach that integrates the latter three separate subsystems above into a joint inference model using a Markov Logic Network (MLN) [Richardson and Domingos, 2006]. Joint models have become popular in NLP recently, because they allow different NLP tasks to be carried out simultaneously and makes it possible for features and constraints to be shared among tasks [Che and Liu, 2010]. Our MLN model can simultaneously exploit contextual information and normalization constraints (e.g. one gene mention can only be normalized to one ID when the mention has not been filtered) to finds the global optimal solution.

2 Methods and Techniques

We embed our AI system into a user's web browser by using the user-script technology based on the GreaseMonkey extension, which enriches the original PubMed search results with semantic annotation generated by our NLP server. Figure 1 shows the system flowchart.

The user must first install our browser extension downloaded from our website, which allows the user to run our user-scripts that make on-the-fly changes to the HTML of the PubMed search results page. When a user visits the PubMed website and invokes a query (Step 1 in Figure 1), the user-script intercepts the returned results (Step 2) and generates an NLP processing request to our NLP server (Step 3). Upon receiving the request, the server forwards it to our text-mining subsystems, which process the intercepted search results and return annotated results back to the user's browser. The user-script then integrates these into the displayed page (Step 4).

2.1 User-scripts

A user-script is a small program written in JavaScript that is automatically run within a web browser when the user accesses pages that match a particular URL. The user-scripts developed in this paper work in-browser only on the PubMed domain.

In order to modify the PubMed HTML pages, the script uses the browser document object model (DOM), which defines the content, structure and style of an HTML document. For example, using the DOM interface, one of our user-scripts can determine which DOM element contains the abstract text returned by PubMed. It can then extract the raw text from the element, and send it to the NLP web server. Upon receiving the processed results, the user-script modifies the appropriate DOM element attribute and the content displayed on the user's browser is updated immediately.

2.2 NLP Web Server

The NLP web server integrates several text-mining systems, such as a sentence splitter, a part-of-speech tagger and a disease and gene mention recognizer. As shown in Figure 1, an abstract sent by the user-script is preprocessed by (1) the sentence splitter to determine the sentence boundaries, (2) the part-of-speech tagger to generate part-of-speech annotations and (3) the gene mention recognizer to generate a list of all gene mention candidates in the abstract (Gene Mention List). After the abstract text is processed and tagged in the above three steps, it is sent to the section categorizer and the GN subsystem.

The function of the section categorizer is to divide a given abstract into section paragraphs. For the section categorization problem, we regard each sentence in an abstract as a token. Each token is associated with a boundary tag, that is the beginning (*B*), inside (*I*) or outside (*O*) of a section, as well as a category tag, *C*, that indicates the category of the section. Therefore, the problem can be formulated as the problem of assigning tags to each token. For example, in *B-C*, *I-C*, where *C* is a section category, *B-* and, *I-* denote, respec-

Table 1. Part-of-speech patterns for tenses

Tense/**Section**	Part-of-speech Pattern
Present/**Background**	VB[ZP]?
Present Perfect/**Purpose,**	VBZ(->VBN){1,2}
Conclusion	VBP(->VBN){1,2}
Past/**Method, Results**	VBD
Past Perfect/**Purpose**	VBD(->VBN){1,2}
Tentative Verbs/**Conclusion**	MD->VB
Modal Auxiliaries/**Conclusion**	MD->VB

tively, the first token and the subsequent token of a section in category *C*. The underlying machine learning model is the conditional random field (CRF). We describe three main features of our model here. Other useful features for the section categorization problem can be found in [Hirohata *et al.*, 2008; Lin *et al.*, 2009].

The first is the "position" feature. The relative position of a sentence in an abstract provides useful information to determine which section it belongs to. We use the following equation to calculate the relative position of a sentence:

$$\text{relative}_{position}(s, S) = 10[{}^{s}/_{S}]$$

where s is the sentence's position in the abstract, and S is the total number of sentences in the abstract. The second is the "tense feature". Weissberg and Buker [1990] suggested that an abstract has five important sections, "Background information", "Purpose/Principal activity", "Methodology", "Results" and "Conclusion", which are often written in specific tenses. For example, the results section is usually in past tense. Based on the part-of-speech information generated by the part-of-speech tagger subsystem, we use seven patterns shown in Table 1 to determine the tense of a given sentence and the corresponding section. These patterns are developed based on [Weissberg and Buker, 1990]. The final feature is the "NE feature", which refers to the named entity information. Since an abstract's title can be treated as a summary of the abstract, and entities in the title often appear in the Results section, we can use co-occurrence ratio of entities in the title and in a given sentence to identify the Results section.

The second dependent system, the GN system, normalizes the gene names in the "Gene Mention List" to their corresponding EntrezGene database IDs. As shown in Figure 1, the GN subsystem includes three components: a candidate ID matcher, a false positive filter and a disambiguation component. Generally speaking, the current top-performing GN systems [Hakenberg *et al.*, 2008; Dai *et al.*, 2010] used the following step-by-step approach. The candidate ID matcher looks up each gene candidate in the "Gene Mention List" in a lexicon of gene names and IDs compiled from EntrezGene. Ideally, after one or several IDs matches have been found for a gene mention, we should be able to treat all these IDs as candidates, and proceed directly to the disambiguation task. However, it is not always the case, because the employed recognizer may generate false positive gene mentions. Therefore, the false-positive filter is employed to decide whether to keep the mention, or to discard it. If the mention is kept, the disambiguation component is then used to select the most appropriate ID for it.

In the following section, we describe an approach that employs an MLN to model the constraints used in the above GN process and show how various decisions can be formulated and combined in an MLN to improve system performance.

2.3 MLN-based Gene Normalization

Markov Logic Network

Markov logic network (MLN) is a joint model which combines first order logic and Markov networks. In first-order logic, formulae consist of four types of symbols: constants, variables, functions, and predicates. *Constants* represent objects in a specific domain (e.g. people: **Joe**, **Sally**, etc.). *Variables* (e.g., *x*, *y*) range over the objects. *Predicates* represent relationships among objects (e.g., *married_to*), or attributes of objects (e.g. *husband*/*wife*). Constants and variables may belong to specific types. An *atom* is a predicate symbol applied to a list of arguments, which may be constants or variables (e.g., *married_to*(*x*, **Joe**)). A *ground atom* is an atom whose arguments are all constants. A *world* is an assignment of truth values to all possible ground atoms. A *knowledge base* (KB) is a partial specification of a world; each atom in it is true, false or unknown.

A Markov network represents the joint distribution of a set of variables $X = (X_1, \dots, X_n) \in \mathcal{X}$ as a product of factors: $P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z}\prod_k f_k(\mathbf{x}_k)$, where each factor f_k is a non-negative function of a subset of the variables $\mathbf{x}_k$, and Z is a normalization constant. The distribution is usually equivalently represented as a log-linear form: $P(\mathbf{X} = \mathbf{x}) = \frac{1}{Z}\exp(\sum_i w_i g_i(\mathbf{x}))$, where the features $g_i(\mathbf{x})$ are arbitrary functions of (a subset of) the variables' states.

An MLN is a set of weighted first-order formulae. Together with a set of constants representing objects in the domain, it defines a Markov network with one variable per ground atom and one feature per ground formula. The probability distribution over possible worlds x is given by $\mathrm{P}(\boldsymbol{X} = x) = \frac{1}{Z}\exp\left(\sum_{i \in F}\sum_{j \in G_i} w_i g_j(x)\right)$ where Z is the partition function, F is the set of all first-order formulae in the MLN, g_j is the set of groundings of the ith first-order formula, and $g_j(x) = 1$ if the jth ground formula is true and $g_j(x) = 0$ otherwise. Markov logic enables us to compactly represent complex models in non-i.i.d. domains. General algorithms for inference and learning in Markov logic are discussed in Richardson and Domingos [2006]. We employ thebeast toolkit[‡] to implement our MLN model. It uses 1-best MIRA online learning method [Crammer and Singer, 2003] for learning weights and employs cutting plane inference [Riedel, 2008] with integer linear programming as its base solver for inference at test time as well as during the MIRA online learning process.

Modeling the GN Disambiguation Process

We use the predicate $isNormalizedTo(i, id)$ to represent the disambiguation component: gene mention i should be nor-

[‡] http://code.google.com/p/thebeast/

Table 2. MLN formulae for disambiguation

$$\exists!\, id.\, isCandidateOf(id, i) \Rightarrow isNormalizedTo(i, id)$$
$$hasChromosomeInfo(i, id, +sd) \Rightarrow isNormalizedTo(i, id)$$
$$\exists!\, id.\, hasPPIPartnerRank(i, id, 1) \wedge hasWord(w) \wedge PPIKeyword(w) \Rightarrow isNormalizedTo(i, id)$$
$$\exists!\, id.\, hasGOTermRank(i, id, 1) \Rightarrow isNormalizedTo(i, id)$$
$$\exists!\, id.\, hasTissueTermRank(i, id, 1) \Rightarrow isNormalizedTo(i, id)$$
$$i < j \wedge isNormalizedTo(i, id) \wedge isCandidateOf(id, j) \Rightarrow isNormalizedTo(j, id)$$
$$isNormalizedTo(i, id_1) \wedge id_1 \neq id_2 \Rightarrow \neg isNormalizedTo(i, id_2)$$

malized to id. The most general assumption is that if a gene mention is mapped to only one ID, it should be normalized to that ID. This is defined as:

Formula 1

$$\exists!\, id.\, isCandidateOf(id, i) \Rightarrow isNormalizedTo(i, id)$$

where $isCandidateOf(id, i)$ represents that the decision of the candidate ID matcher: identifier candidate id is a candidate of the gene mention.

Note that, in our formulae, we refer to a gene mention by its order in the article (e.g., the ith gene mention) for several reasons. One, not all names can be found in the training data. Secondly, even if two gene mentions have the same name string, they may normalize to different IDs.

If a gene mention has two or more candidate IDs, we must determine which is more appropriate through disambiguation processing. We implement most of the rules employed by [Lai *et al.*, 2009; Dai *et al.*, 2010]. For example, we define the predicate $hasChromosomeInfo(i, id, sd)$ to indicate that the chromosome location information of the ith gene mention, which has the identifier id as its candidate ID, can be found in the surrounding text in the range sd. Applying this predicate to the sentence:

"The human **UBQLN3** gene was mapped to the *11p15* region of chromosome 11.",

The mention UBQLN3 must be normalized to the EntrezGene ID 50613 because 50613's chromosome location, *11p15*, is found in the context. The formula describing the relation of $hasChromosomeInfo$ and $isNormalizedTo$ is defined as follows:

$$hasChromosomeInfo(i, id, +sd) \Rightarrow isNormalizedTo(i, id)$$

Here, we can see that there is an additional parameter, $+sd$, in the predicate $hasChromosomeInfo$. sd, indicating where the chromosome information corresponding to id locates, has two possible values: 0 indicates the id's chromosome information is located in the same sentence as i. Otherwise, sd is 1. The "+" notation in the above formula indicates that the MLN must learn a separate weight for each grounded variable (sd). For example, $hasChromosomeInfo(i, id, 0)$ and $hasChromosomeInfo(i, id, 1)$ are given two different weights in our MLN model after training. Table 2 briefly summarizes the main formulae defined for $isNormalizedTo$.

Modeling the False Positive Filtering

For GN, false positives can be classified into two types: those that do not belong to any entity class, and those that belong to classes that are not the curation target, e.g. DNA polymerases, or protein families. To capture the filtering concept in our model, we define the predicate $ShouldBeNormalized(i)$, which indicates that the gene mention i of the article should be normalized to an ID. In our model, for the ith entity, if the possible world $ShouldBeNormalized(i)$ is false, the entity is considered as a false positive.

Table 3. MLN formulae for false positive filtering

$$hasGeneName(i, +n) \Rightarrow ShouldBeNormalized(i)$$
$$hasFirstWord(i, +w) \wedge SpeciesTerm(+w) \Rightarrow ShouldBeNormalized(i)$$
$$hasPrecedingWord(i, +w, 1) \wedge SpeciesTerm(+w) \Rightarrow ShouldBeNormalized(i)$$
$$hasFollowingWord(i, +w, 1) \Rightarrow ShouldBeNormalized(i)$$
$$containsMoreSpecificMentions(i) \Rightarrow \neg ShouldbeNormalized(i)$$
$$isNormalizedTo(i, id) \Rightarrow ShouldBeNormalized(i)$$

The first formula containing $ShouldBeNormalized$ is associated with different weights by considering the grounded gene name n:

$$hasGeneName(i, +n) \Rightarrow ShouldBeNormalized(i)$$

The other formulae determine whether i is a true gene mention or not by checking i's context. For example:

$$hasFirstWord(i, +w) \wedge SpeciesTerm(+w) \Rightarrow ShouldBeNormalized(i)$$

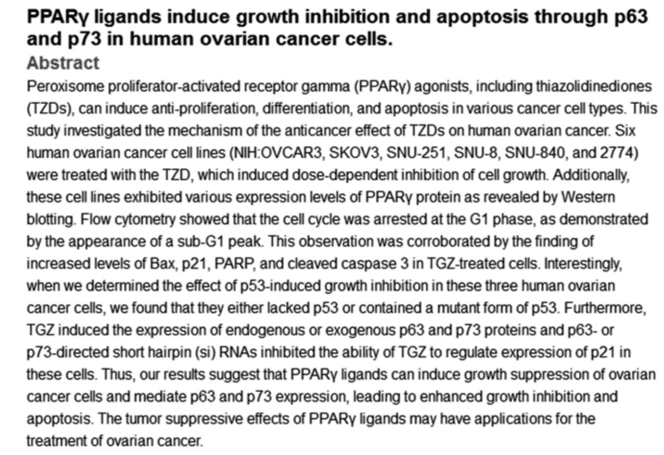
implies that a certain gene mention i's suitability for normalizing depends on whether or not i's first word is a certain species keyword. Table 3 summarizes the main formulae defined for $ShouldBeNormalized$.

3 Results

3.1 User Scenarios

Figure 2 shows an example of the uncategorized biomedical abstract returned by PubMed search. As one can see, it is difficult to quickly differentiate the background, methods, results and conclusion sections because the monochromatic text is squeezed together into one long paragraph.

Figure 3 shows the same search results marked up by our embedded AI system. As you can see, the uncategorized abstract is automatically sectioned by our system into OBJECTIVE, METHOD, RESULT, and CONCLUSION. In Figure 3, the gene mentions in the displayed results are also marked in different colors. When users move their mouse

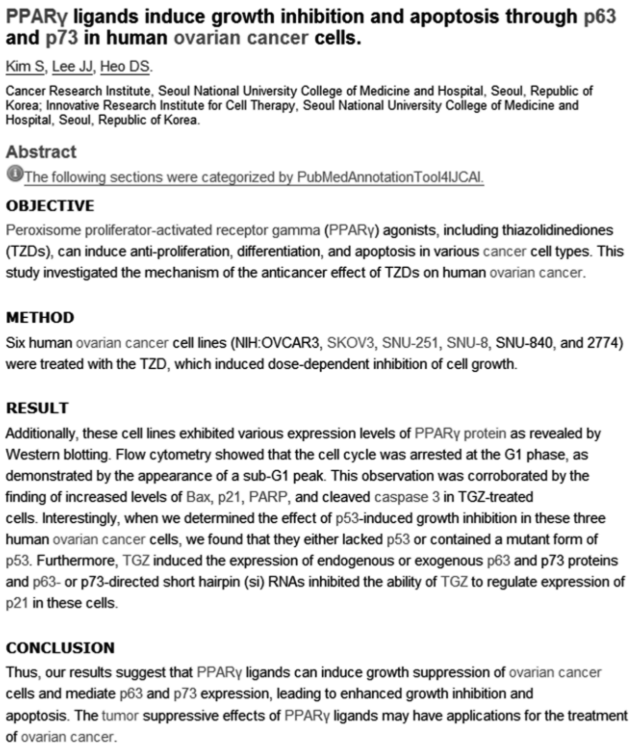


PPARγ ligands induce growth inhibition and apoptosis through p63 and p73 in human ovarian cancer cells.

Abstract

Peroxisome proliferator-activated receptor gamma (PPARγ) agonists, including thiazolidinediones (TZDs), can induce anti-proliferation, differentiation, and apoptosis in various cancer cell types. This study investigated the mechanism of the anticancer effect of TZDs on human ovarian cancer. Six human ovarian cancer cell lines (NIH:OVCAR3, SKOV3, SNU-251, SNU-8, SNU-840, and 2774) were treated with the TZD, which induced dose-dependent inhibition of cell growth. Additionally, these cell lines exhibited various expression levels of PPARγ protein as revealed by Western blotting. Flow cytometry showed that the cell cycle was arrested at the G1 phase, as demonstrated by the appearance of a sub-G1 peak. This observation was corroborated by the finding of increased levels of Bax, p21, PARP, and cleaved caspase 3 in TGZ-treated cells. Interestingly, when we determined the effect of p53-induced growth inhibition in these three human ovarian cancer cells, we found that they either lacked p53 or contained a mutant form of p53. Furthermore, TGZ induced the expression of endogenous or exogenous p63 and p73 proteins and p63- or p73-directed short hairpin (si) RNAs inhibited the ability of TGZ to regulate expression of p21 in these cells. Thus, our results suggest that PPARγ ligands can induce growth suppression of ovarian cancer cells and mediate p63 and p73 expression, leading to enhanced growth inhibition and apoptosis. The tumor suppressive effects of PPARγ ligands may have applications for the treatment of ovarian cancer.

Figure 2. Query result for the abstract (PMID:21329655) from PubMed search.



PPARγ ligands induce growth inhibition and apoptosis through p63 and p73 in human ovarian cancer cells.

Kim S, Lee JJ, Heo DS.

Cancer Research Institute, Seoul National University College of Medicine and Hospital, Seoul, Republic of Korea; Innovative Research Institute for Cell Therapy, Seoul National University College of Medicine and Hospital, Seoul, Republic of Korea.

Abstract

The following sections were categorized by PubMedAnnotationTool4IJCAI.

OBJECTIVE

Peroxisome proliferator-activated receptor gamma (PPARγ) agonists, including thiazolidinediones (TZDs), can induce anti-proliferation, differentiation, and apoptosis in various cancer cell types. This study investigated the mechanism of the anticancer effect of TZDs on human ovarian cancer.

METHOD

Six human ovarian cancer cell lines (NIH:OVCAR3, SKOV3, SNU-251, SNU-8, SNU-840, and 2774) were treated with the TZD, which induced dose-dependent inhibition of cell growth.

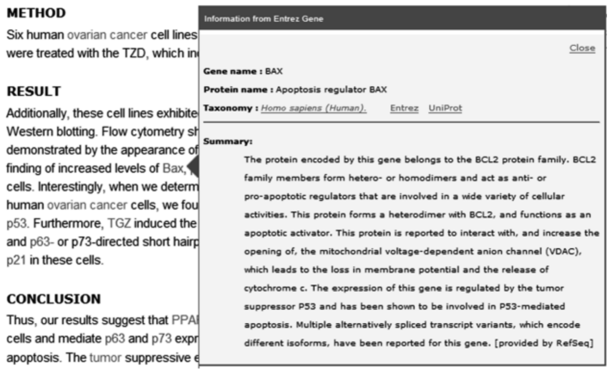
RESULT

Additionally, these cell lines exhibited various expression levels of PPARγ protein as revealed by Western blotting. Flow cytometry showed that the cell cycle was arrested at the G1 phase, as demonstrated by the appearance of a sub-G1 peak. This observation was corroborated by the finding of increased levels of Bax, p21, PARP, and cleaved caspase 3 in TGZ-treated cells. Interestingly, when we determined the effect of p53-induced growth inhibition in these three human ovarian cancer cells, we found that they either lacked p53 or contained a mutant form of p53. Furthermore, TGZ induced the expression of endogenous or exogenous p63 and p73 proteins and p63- or p73-directed short hairpin (si) RNAs inhibited the ability of TGZ to regulate expression of p21 in these cells.

CONCLUSION

Thus, our results suggest that PPARγ ligands can induce growth suppression of ovarian cancer cells and mediate p63 and p73 expression, leading to enhanced growth inhibition and apoptosis. The tumor suppressive effects of PPARγ ligands may have applications for the treatment of ovarian cancer.

Figure 3. Query results after section categorization



METHOD

Six human ovarian cancer cell lines
were treated with the TZD, which in

RESULT

Additionally, these cell lines exhibite
Western blotting. Flow cytometry sh
demonstrated by the appearance of
finding of increased levels of Bax,
cells. Interestingly, when we determ
human ovarian cancer cells, we fou
p53. Furthermore, TGZ induced the
and p63- or p73-directed short hairp
p21 in these cells.

CONCLUSION

Thus, our results suggest that PPAI
cells and mediate p63 and p73 expr
apoptosis. The tumor suppressive e

Information from Entrez Gene

Close

Gene name : BAX

Protein name : Apoptosis regulator BAX

Taxonomy : *Homo sapiens (Human).* Entrez UniProt

Summary:

The protein encoded by this gene belongs to the BCL2 protein family. BCL2 family members form hetero- or homodimers and act as anti- or pro-apoptotic regulators that are involved in a wide variety of cellular activities. This protein forms a heterodimer with BCL2, and functions as an apoptotic activator. This protein is reported to interact with, and increase the opening of, the mitochondrial voltage-dependent anion channel (VDAC), which leads to the loss in membrane potential and the release of cytochrome c. The expression of this gene is regulated by the tumor suppressor P53 and has been shown to be involved in P53-mediated apoptosis. Multiple alternatively spliced transcript variants, which encode different isoforms, have been reported for this gene. [provided by RefSeq]

Figure 4. Pop-up summary of a recognized gene

cursors over a recognized gene, a brief pop-up summary will be displayed as shown in Figure 4. Furthermore, the recognized genes hyperlink to EntrezGene pages; users can click on recognized genes to open a new browser window containing more detailed information.

3.2 Performance of Text-Mining Subsystems

Section Categorization Performance

Since there are no publicly available section categorization corpora, we constructed a corpus using the following procedure. Firstly, we compiled dozens of likely section headings from [Hirohata *et al.*, 2008] into a list, *SH*. Secondly, we searched PubMed using the keyword queries "hypertension" and "hypertension and (gene or DNA or RNA)" and compiled the approximately thirteen thousand results into a corpus after filtering out un-sectioned abstracts. Thirdly, we designed and implemented a program to remove section tags

Table 4. The performance of section categorization.

Feature Set	P (%)	R (%)	F (%)
(1) Baseline (Position)	87.08	93.06	89.97
(2) (1) +Tense+NE features	92.87	92.29	92.58

from the corpus abstracts based on headings in *SH*. Finally, our in-lab biologists manually checked section boundaries in the collected abstracts. The compiled corpus can be downloaded from our website.

To evaluate the performance of our section categorizer, we applied 3-fold cross-validation using this corpus. Table 4 shows the evaluation results. As you can see, the combination of the Position feature with Tense and NE features can improve the precision by 5.79% without reducing the recall too much and leads to an improved F-score of 92.58%. The results show that an AI system's performance could be improved by integrating other text-mining systems' prediction results.

GN Performance

We used the dataset provided by the BioCreAtIvE II GN task [Morgan *et al.*, 2008] to evaluate the GN performance. The test corpus contains 785 gene IDs among 262 abstracts. Detail s on these two independent training and test datasets can be found at http://biocreative.sourceforge.net.

We compare our MLN-based GN system that performs joint learning and inference with the system released by Lai et al. [2009][§]. Lai's system is based on the step-by-step GN approach. In addition, we use the features equivalent to the formulae shown in Table 2 to implement the false positive filtering component for Lai's system, which is based on the maximum entropy model. Table 5 compares the performance of our MLN-based system and Lai's system.

The first row (No disambiguation/F.1) show the performance of Lai' system without applying their disambiguation approach. In this configuration, all mentions with only one candidate ID were directly treated as answers, and entities with more than one candidate ID were discarded. Our MLN-based model can simulate the result when only Formula 1 (F.1) is applied. Rows of (a, b) compare our MLN-based disambiguation approach, which uses the F.1 and all formulae defined in Table 2, with Lai's system. The (d) employs the false positive maximum entropy model to further filter out false positives. Our MLN-based GN system (c) achieves the same goal by adding on (a) with formulae defined in Table 3.

In summary, we observe that the MLN-based disambiguation method outperforms the compared method by 1.5%. By adding the false positive filtering formula, our MLN-based

Table 5. The performance of GN.

Config.	P	R	F	Diff
No disambiguation/**F.1**	77.3	71.4	74.2	0
(a) MLN-based/Table 2	86.1	83.0	**84.5**	+10.3
(b) Lai's System	82.6	83.4	83.0	+8.8
(c) MLN-based/(<u>a</u>)+Table 3	89.7	81.9	**85.6**	+11.33
(d) (b) + false positive filtering model	84.7	83.4	84.0	+9.7

[§] https://sites.google.com/site/potinglai/downloads

Table 6. The performance of false positive filter.

Config.	P	R	F
(a) MLN model (Joint)	66.4	98.9	**79.5**
(b) Maximum entropy model	62.4	99.9	76.8

GN system performance is boosted by 1.03%. Table 6 further compares the performance of false positive filter in the two different models. As you can see that the joint model achieves better F-score. The results show that by using the joint model, different subsystems can help each other to improve their performance.

4 Conclusions

In this paper, we have presented an AI system embedded in a web browser, which adds functionality to PubMed, the most popular biomedical search service. The AI system seamlessly integrates the NLP features of our web server into PubMed's online search interface via the user-script technology. With our system, researchers using a supported browser can now enjoy advanced text mining features on PubMed's website without having to regularly visit a third-party site. The system's page mark-up can help researchers sort through abstracts efficiently, quickly focus on key genes, and can provide additional background information. Furthermore, our distributed web server takes on the text processing costs, freeing up users' computational resources.

In addition, we show that an AI system's performance could be improved by integrating other text-mining systems' prediction results. We also describe a novel approach that employs MLN to model the constraints and decisions in the GN task and show that integrating different subsystems into a simultaneous process can achieve better performance of predicting gene mentions and their corresponding IDs in contrast to a step-by-step approach, which identifies mentions first and then normalizes them to IDs.

Acknowledgments

This research was supported in part by the National Science Council under grant NSC99-3112-B-001-005, NSC 98-3112-B-001-009 and NSC97-3112-B-001-011, and the Academia Sinica Investigator Award. We especially thank the IJCAI reviewers for their valuable comments.

References

[Che and Liu, 2010] Che, W. and T. Liu, 2010. Jointly modeling WSD and SRL with markov logic. pages 161-169.

[Crammer and Singer, 2003] Crammer, K. and Y. Singer, 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3: 951-991.

[Dai *et al.*, 2008] Dai, H.-J., C.-H. Huang, R.T.K. Lin, R.T.-H. Tsai and W.-L. Hsu, 2008. Biosmile Web Search: A web application for annotating biomedical entities and relations. *Nucl. Acids Res.*, 36(Web Server issue): W390-W398.

[Dai *et al.*, 2010] Dai, H.-J., P.-T. Lai and R.T.-H. Tsai, 2010. Multistage gene normalization and svm-based ranking for protein interactor extraction in full-text articles. *IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS*, 7(3): 412-420.

[Hakenberg *et al.*, 2008] Hakenberg, J., C. Plake, R. Leaman, M. Schroeder and G. Gonzalez, 2008. Inter-species normalization of gene mentions with gnat. *Bioinformatics*, 24(16): 126-132.

[Hirohata *et al.*, 2008] Hirohata, K., N. Okazaki, S. Ananiadou and M. Ishizuka, 2008. Identifying sections in scientific abstracts using conditional random fields.

[Hoffmann and Valencia, 2005] Hoffmann, R. and A. Valencia, 2005. Implementing the iHOP concept for navigation of biomedical literature. *Bioinformatics*, 21(2): 252-258.

[Lai *et al.*, 2009] Lai, P.-T., Y.-Y. Bow, C.-H. Huang, H.-J. Dai, R.T.-H. Tsai and W.-L. Hsu, 2009. Using contextual information to clarify gene normalization ambiguity. pages 1-5.

[Lin *et al.*, 2009] Lin, R.T.K., H.-J. Dai, Y.-Y. Bow, J.L.-T. Chiu and R.T.-H. Tsai, 2009. Using conditional random fields for result identification in biomedical abstracts *Integrated Computer-Aided Engineering*, 16(4): 339-352.

[Mons *et al.*, 2008] Mons, B., M. Ashburner, C. Chichester, E. van Mulligen, M. Weeber, J. den Dunnen, et al., 2008. Calling on a million minds for community annotation in wikiproteins. *Genome Biology*, 9(5): R89.

[Morgan *et al.*, 2008] Morgan, A., Z. Lu, X. Wang, A. Cohen, J. Fluck, P. Ruch, A. Divoli, K. Fundel, R. Leaman, J. Hakenberg, C. Sun, H.-h. Liu, R. Torres, M. Krauthammer, W. Lau, H. Liu, C.-N. Hsu, M. Schuemie, K.B. Cohen and L. Hirschman, 2008. Overview of biocreative ii gene normalization. *Genome Biology*, 9(Suppl 2): S3.

[Pafilis *et al.*, 2009] Pafilis, E., S.I. O'Donoghue, L.J. Jensen, H. Horn, M. Kuhn, N.P. Brown and R. Schneider, 2009. Reflect: Augmented browsing for the life scientist. *Nat Biotech*, 27(6): 508-510.

[Richardson and Domingos, 2006] Richardson, M. and P. Domingos, 2006. Markov logic networks. *Machine Learning*, 62: 107-136.

[Riedel, 2008] Riedel, S., 2008. Improving the accuracy and efficiency of map inference for markov logic. *Proceedings of UAI 2008*.

[Weissberg and Buker, 1990] Weissberg, R. and S. Buker, 1990. Writing up research: Experimental research report writing for students of english. Prentice Hall Regents.

[Willighagen *et al.*, 2007] Willighagen, E., N. O'Boyle, H. Gopalakrishnan, D. Jiao, R. Guha, C. Steinbeck and D. Wild, 2007. Userscripts for the life sciences. *BMC Bioinformatics*, 8(1): 487.