Technical Report

The KNIME Text Processing Feature:

An Introduction

Dr. Killian Thiel Killian.Thiel@uni-konstanz.de
Dr. Michael Berthold Michael.Berthold@uni-konstanz.de

Table of Contents

Introduction

The KNIME[1] Text Processing feature was designed and developed to read and process textual data, and transform it into numerical data (document and term vectors) in order to apply regular KNIME data mining nodes (e.g. for clustering and classification). This feature allows for the parsing of texts available in various formats (e.g. Xml, Microsoft Word or PDF and the internal representation of documents and terms) as KNIME data cells stored in a data table. It is possible to recognize and tag different kinds of named entities such as names of persons and organizations, genes and proteins or chemical compounds, thus enriching the documents semantically. Furthermore, documents can be filtered (e.g. by stop word or named entity filters), stemmed by stemmers for various languages and preprocessed in many other ways. Frequencies of words can be computed, keywords can be extracted, and documents can be visualized (e.g.tag clouds). To apply regular KNIME nodes to cluster or classify documents, they can be transformed into numerical vectors.

This paper is organized as follows: In the next section, the philosophy and usage of the feature is described; meaning which nodes can be applied in which order, what needs to de done and what can be done and in which step. In Section "Data Structures" the different structures of data tables are described as well as the new KNIME data types. The last section concludes this document.

Philosophy

In this section, the philosophy of the Text Processing feature is explained, (i.e. which nodes of which categories have to or can be applied and in which order). The order of the nodes is important since they require specific structural specifications of the input data tables, e.g. the “BoW creator” node. Creating a bag of words requires a data table with one column containing *DocumentCells*. The output data table of this node represents a bag of words consisting of two columns; one containing *DocumentCells* and the other containing *TermCells*. This node will only work (configure) properly with an input data table consisting of a column of *DocumentCells*. The “BoW creator” node, and all other nodes of the Text Processing feature require input data tables of specific structure and create output data tables of specific structure. Consequently, certain types of nodes can only be used in a certain order. This order or configuration is the basic philosophy of the KNIME Text Processing feature.

To process texts with the KNIME Text Processing feature, usually six different steps need to be accomplished. These steps are:

- IO
- Enrichment
- Preprocessing
- Frequencies
- Transformation
- Visualization

For each of these steps there exists a category folder in the Text Processing node repository (except for the visualization nodes, which are located in the “Misc” category).

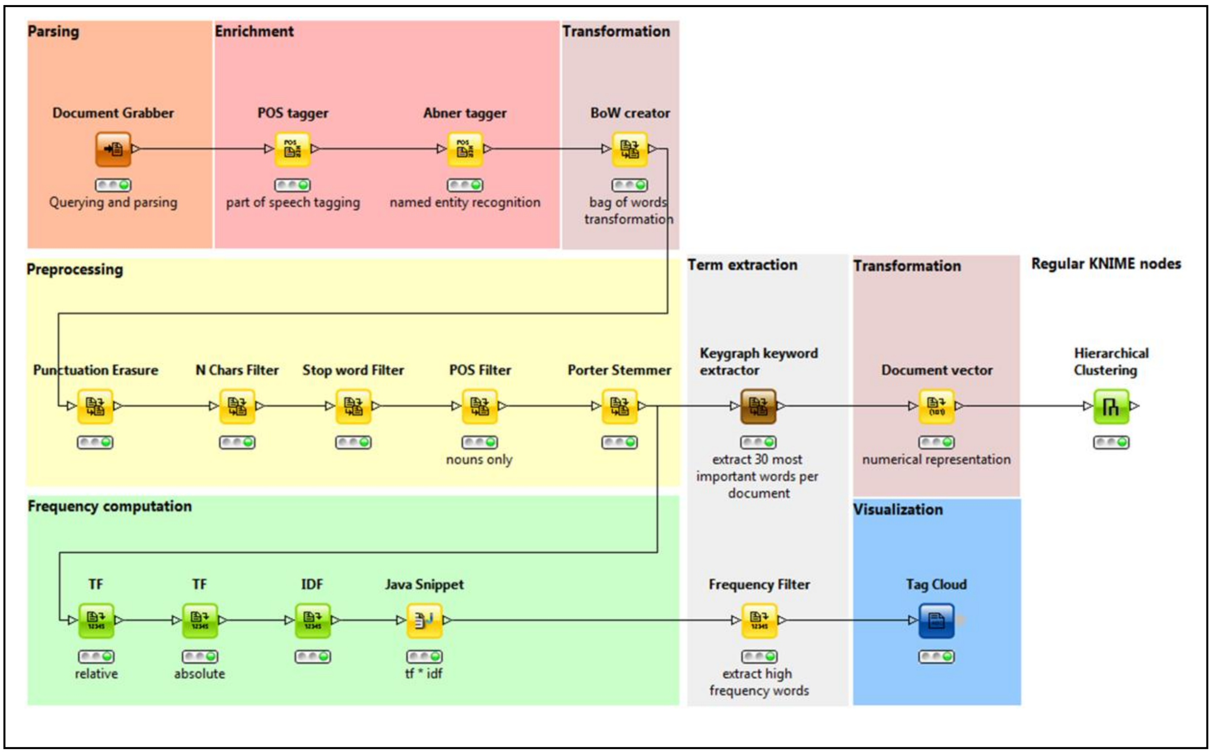
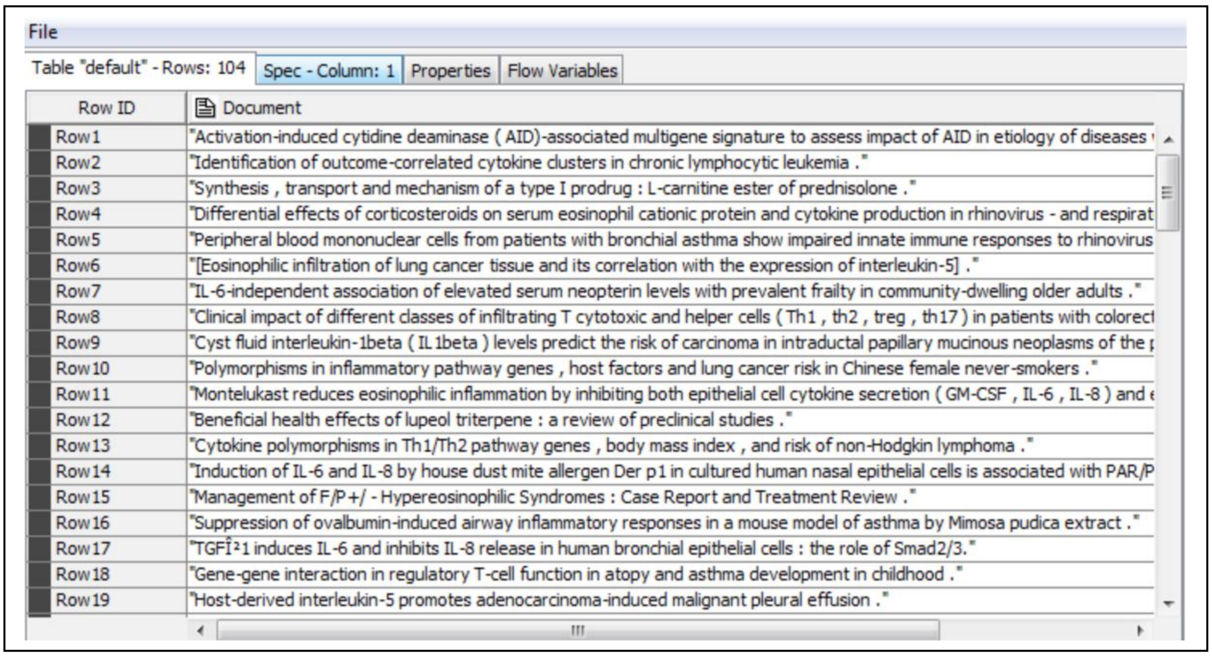


Figure 1: An example workflow illustrating the basic philosophy and order of Text Processing nodes.

IO

First, the text documents have to be parsed. The text and the structure, if available, have to be extracted and represented in a data structure the Text Processing nodes are able to handle. A document is represented internally by the KNIME data type *DocumentCell* and *DocumentValue*. More details about these data types can be found in Section "Data Structures". Each parser node reads text documents of a certain format and creates a *DocumentCell* for each document.

All provided parser nodes are contained in the IO subcategory of the Text Processing category. For formatss such as DML, SDML, PubMed (XML format), PDF, Microsoft Word, and flat files, parser nodes are available. The output of all parser nodes is a data table consisting of one column with *DocumentCells*. An example of such a data table is illustrated in Figure 2. The icon, shown in the header of the second column, indicates that the type of the data contained by this column is the document data. Such a list of documents can then be used as input by all nodes of the enrichment category.



Row ID	Document
Row1	"Activation-induced cytidine deaminase (AID)-associated multigene signature to assess impact of AID in etiology of diseases
Row2	"Identification of outcome-correlated cytokine clusters in chronic lymphocytic leukemia ."
Row3	"Synthesis , transport and mechanism of a type I prodrug : L-carnitine ester of prednisolone ."
Row4	"Differential effects of corticosteroids on serum eosinophil cationic protein and cytokine production in rhinovirus - and respirat
Row5	"Peripheral blood mononuclear cells from patients with bronchial asthma show impaired innate immune responses to rhinovirus
Row6	"[Eosinophilic infiltration of lung cancer tissue and its correlation with the expression of interleukin-5] ."
Row7	"IL-6-independent association of elevated serum neopterin levels with prevalent frailty in community-dwelling older adults ."
Row8	"Clinical impact of different classes of infiltrating T cytotoxic and helper cells (Th1 , th2 , treg , th17) in patients with colorect
Row9	"Cyst fluid interleukin-1beta (IL1beta) levels predict the risk of carcinoma in intraductal papillary mucinous neoplasms of the
Row10	"Polymorphisms in inflammatory pathway genes , host factors and lung cancer risk in Chinese female never-smokers ."
Row11	"Montelukast reduces eosinophilic inflammation by inhibiting both epithelial cell cytokine secretion (GM-CSF , IL-6 , IL-8) and
Row12	"Beneficial health effects of lupeol triterpene : a review of preclinical studies ."
Row13	"Cytokine polymorphisms in Th1/Th2 pathway genes , body mass index , and risk of non-Hodgkin lymphoma ."
Row14	"Induction of IL-6 and IL-8 by house dust mite allergen Der p1 in cultured human nasal epithelial cells is associated with PAR/P
Row15	"Management of F/P+/ - Hypereosinophilic Syndromes : Case Report and Treatment Review ."
Row16	"Suppression of ovalbumin-induced airway inflammatory responses in a mouse model of asthma by Mimosa pudica extract ."
Row17	"TGFÎ²1 induces IL-6 and inhibits IL-8 release in human bronchial epithelial cells : the role of Smad2/3."
Row18	"Gene-gene interaction in regulatory T-cell function in atopy and asthma development in childhood ."
Row19	"Host-derived interleukin-5 promotes adenocarcinoma-induced malignant pleural effusion ."

Figure 2: A data table containing a list of PubMed documents, one in each row.

The DML and SDML formats are XML based formats which represent texts in a structured way. Texts available in other XML based formats can be transformed into SDML via Xslt and the usage of the XML nodes provided by the KNIME XML feature.

Textual data which is stored in csv format can be red by the "File Reader" node. All fields of the documents, such as author, title, abstract, and full text need to separated appropriately and stored in a corresponding column containing *StringCells*. Afterwards the node "Strings to Documents" need to be applied, which creates a *DocumentCell* for each row. In the dialog of the node it needs to be specified which column contains the data for which field of the document. Thereby the "Strings to Documents" node provides an easy way to convert textual data from csv files, Microsoft Excel files, or also data bases into a list of documents.

Enrichment and Tagging

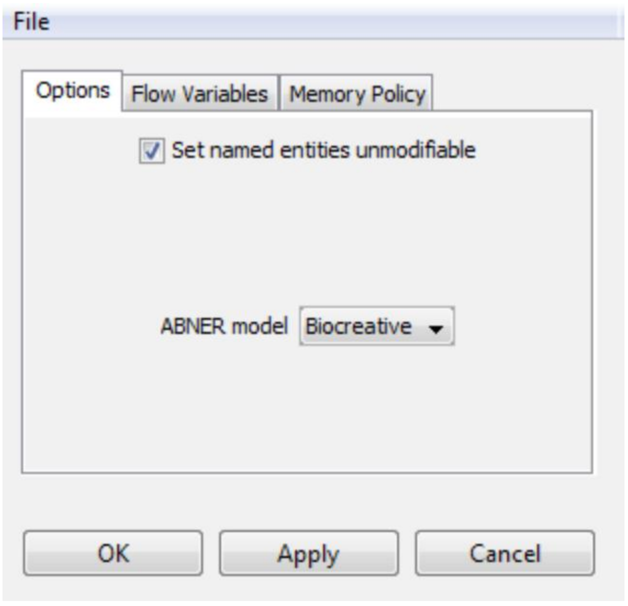
In the Enrichment step, semantic information is added by named entity recognition and tagging. The Enrichment category contains nodes which assign part of speech tags and recognize standard named entities. Examples of these are names of persons, organizations, or locations, biomedical named entities such as names of genes or proteins and chemical structures. All enrichment nodes require an input data table containing exactly one column of *DocumentCells*. The output data table consists of one column of *DocumentCells* as well, whereas the recognized entities in the documents have been tagged.

Each recognized named entity is assigned a tag value, (e.g. person and a tag type, e.g. NE for the named entity). The tag type represents the domain, or type of a tagger, (e.g. biomedical named entities or chemical named entities). The value represents a particular characteristic in that domain, (e.g. gene in the biomedical field). The "POS tagger" node, for example, assigns part of speech tags. The assigned tag type is POS and the values are those of the Penn Treebank1 tag set. German texts can be tagged with the "Stanford tagger" node, with the STTS2 tag set. Each tagger usually assigns tags of its own domain and thus uses its own tag type and set of values. Based on these tag types and values, filtering can be applied afterwards so that the named entities can be extracted and visualized.

To identify and tag standard named entities, (e.g. persons or organizations), the "OpenNLP NE tagger" node, which is based on the OpenNLP project3, can be used. For biomedical entities, (e.g. gene and protein names) the “Abner tagger” node, based on[2], can be used. For chemical compounds the “Oscar tagger” node, based on the Oscar chemical named entity recognizer framework4, can be used. Additionally, a “Dictionary tagger” is provided by the Text Processing feature, which allows for the specification of a dictionary of named entities to identify.

Unmodifiability

Named entities can be set to *unmodifiable* in order to prevent them from being separated, manipulated or filtered by subsequent nodes of the preprocessing category in the workflow (e.g. stemmer). Usually recognized named entities, (e.g. gene names), should not be filtered or stemmed by subsequent nodes in the workflow. To avoid their manipulation of preprocessing nodes, these terms are set *unmodifiable* by the corresponding tagger node (by default). In the dialog of each tagger node it can be specified whether recognized named entities are set *unmodifiable* or not. This is shown in Figure 3.



☑ Set named entities unmodifiable

ABNER model: Biocreative

Figure 3: The dialog of the Abner tagger node with a check box to set the recognized named entities unmodifiable.

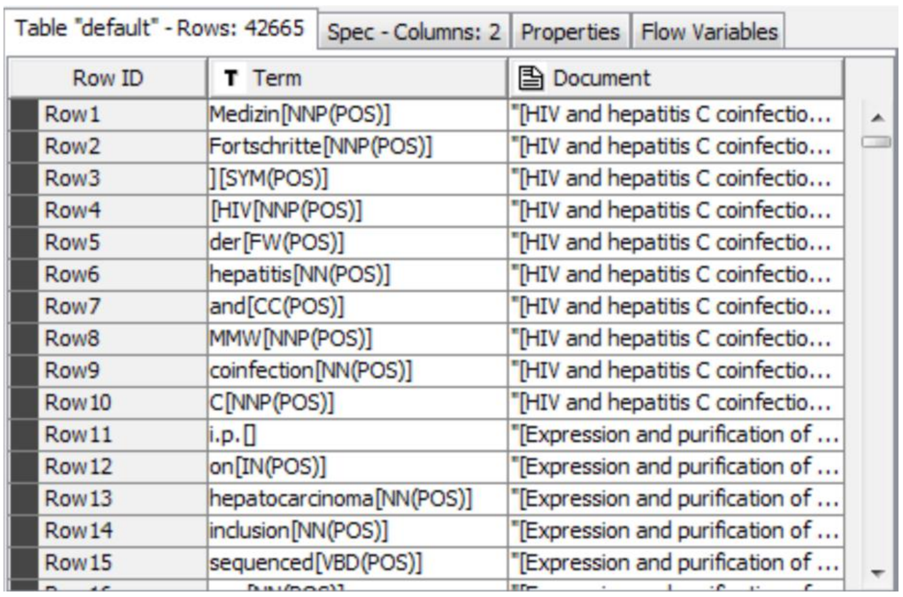
Tagger Conflicts

If two named entity recognizers are applied one after the other, the latter will overwrite the tagging of the former in case of conflicts. For example, if first the “Abner tagger” node recognized “interleukin 7” as a named entity, and second the “Dictionary tagger” node recognizes “interleukin” as a named entity (based on a specified dictionary), than the previously recognized term “interleukin 7” is split up and “interleukin” (without 7) is tagged as a named entity.

Preprocessing

In the preprocessing step, terms are filtered and manipulated in order to get rid of terms that do not contain content. Terms such as stop words, numbers, punctuation marks or very small words are filtered. Terms are also filtered to remove endings based on declination or conjugation by applying stemming. Usually the bag of words (bow) is cleaned in the preprocessing step and only those terms are left over which are used afterwards to create a numerical vector or that are visualized in a tag cloud.

Before nodes of the preprocessing category can be applied on documents, a bag of words has to be created by the usage of the "BoW creator" node. This node transforms a data table consisting of a column containing *DocumentCells* into a bag of words, which is a data table consisting of a column containing *DocumentCells* and a column containing *TermCells*. Figure 4 illustrates a bag of words data table. The BoW data table structure is explained more detailed in Section "Data Table Structures".



Row ID	Term	Document
Row1	Medizin[NNP(POS)]	"[HIV and hepatitis C coinfectio...
Row2	Fortschritte[NNP(POS)]	"[HIV and hepatitis C coinfectio...
Row3	][SYM(POS)]	"[HIV and hepatitis C coinfectio...
Row4	[HIV[NNP(POS)]	"[HIV and hepatitis C coinfectio...
Row5	der[FW(POS)]	"[HIV and hepatitis C coinfectio...
Row6	hepatitis[NN(POS)]	"[HIV and hepatitis C coinfectio...
Row7	and[CC(POS)]	"[HIV and hepatitis C coinfectio...
Row8	MMW[NNP(POS)]	"[HIV and hepatitis C coinfectio...
Row9	coinfection[NN(POS)]	"[HIV and hepatitis C coinfectio...
Row10	C[NNP(POS)]	"[HIV and hepatitis C coinfectio...
Row11	i.p.[]	"[Expression and purification of ...
Row12	on[IN(POS)]	"[Expression and purification of ...
Row13	hepatocarcinoma[NN(POS)]	"[Expression and purification of ...
Row14	inclusion[NN(POS)]	"[Expression and purification of ...
Row15	sequenced[VBD(POS)]	"[Expression and purification of ...

Figure 4: A bag of words data table which consists of one column containing TermCells and one column containing DocumentCells.

Besides the regular preprocessing nodes such as stemming, stop word filtering, etc., there are various other preprocessing nodes available in the Text Processing feature to manipulate and filter terms. For each tag type (tagger) there is a corresponding filter node that filters terms with certain tag values assigned. The tag values can be specified in the dialog of the nodes. The "Stanford tagger" node for example assigns part of speech tags of the STTS tag set to German texts. Correspondingly the "STTS filter" node filters terms which have certain STTS tags assigned. This combination of tagging and filtering allows for a powerful identification and extraction for named entities of different types. It is for example easily possible to identify and extract gene names from PubMed articles and visualize them in a tag cloud.

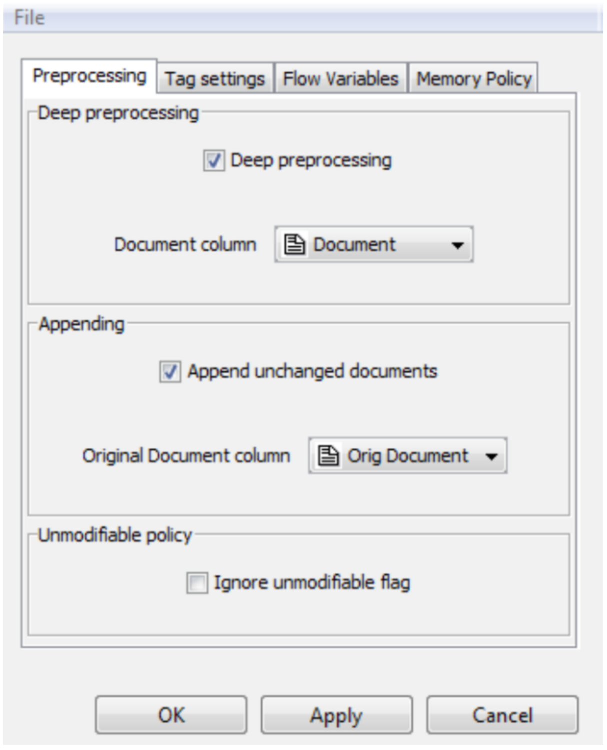
Other very powerful preprocessing nodes are the "RegEx Filter" node and the "Replacer" node which are both based on regular expressions that can be specified in the dialog of the nodes. The "Snowball Stemmer" node allows for the stemming of texts in many languages such as English, German, Spanish and Italian. The "Dict Replacer" node replaces certain terms which are specified in a dictionary with certain other terms, which are specified in the dictionary also. This node allows, for example, a replacement of certain names by specified synonyms.

Preprocessing Dialog

The dialogs of all preprocessing nodes contain a preprocessing tab, shown in Figure 5, in which three settings can be specified:

- Deep preprocessing: The preprocessing of a term (e.g. stemming), is on the one hand applied on each term contained in the term column of the bag of words data table. On the other hand, it can be applied on the terms contained in the documents of the document column, which is called *deep preprocessing*. This is useful if, for example, after preprocessing frequencies have to be computed. If the terms are only preprocessed in the term column but not in the corresponding document itself, the preprocessed term cannot be found in the document afterwards and thus not be counted. Deep preprocessing is applied by default.

- Appending: If deep preprocessing is applied, terms in the documents are preprocessed, (i.e. filtered) and manipulated as well. Sometimes it can be useful to keep the original document. This option can be selected, which is set by default. The original, unchanged documents are appended as additional columns in the bag of words data table.
- Unmodifiable policy: By default all preprocessing nodes do not apply preprocessing on terms that have been set unmodifiable beforehand by a certain tagger node of the Enrichment category. This unmodifiable flag can be ignored.



Deep preprocessing

[✓] Deep preprocessing

Document column: Document

Appending

[✓] Append unchanged documents

Original Document column: Orig Document

Unmodifiable policy

[ ] Ignore unmodifiable flag

Figure 5: The preprocessing tab of the "POS filter" node, with the Deep preprocessing check box, the Appending check box, and the Unmodifiable policy check box.

Frequencies

After preprocessing is finished, frequencies of terms in documents and the complete corpus can be computed. The Text Processing feature provides nodes for the computation of the most famous frequency measures, (i.e. term frequency (tf) “TF” node and inverse document frequency (idf) “IDF” node). In addition to the inverse document frequency, the “ICF” node is available to determine the inverse category frequency (icf), which is analogue to “IDF” but based on categories. Based on the computed frequencies, filtering can be applied by the “Frequency Filter” node in order to keep the high frequent terms. It is possible to specify a range of frequency values for which terms are kept, or to specify an amount k of terms to keep, which are the k terms with the highest frequencies.

Besides extracting keywords based on their tf or idf values, other nodes can be applied such as the “Chi-square keyword extractor” node as well using a chi-square measure to score the relevance[3] or the “Keygraph keyword extractor” node using a graph representation of documents to find keywords[4].

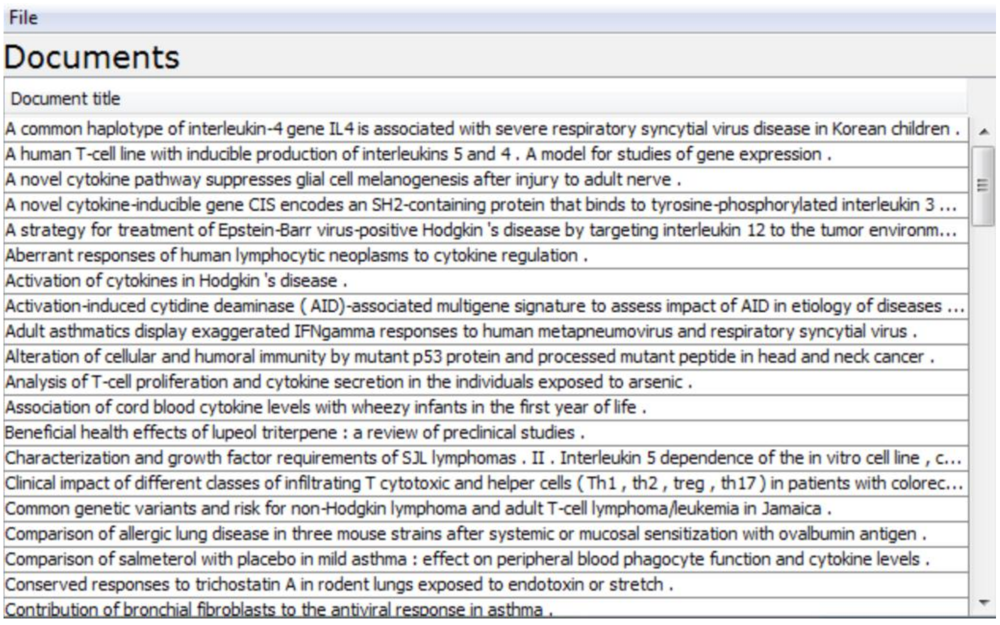
Transformation

Before regular KNIME nodes can be applied on the documents, the textual data has to be transformed into numerical data. This can be done by the "Document vector" or "Term vector" node. These nodes create a binary or numerical vector representation for each term or document, based on the filtered bag of words input data table. The vector representation can then be used by standard data mining methods, such as clustering or classification nodes. Usually the transformation into numerical vectors (or visualization) is the final step which is done by nodes of the Text Processing feature.

In addition to the "Term vector" and "Document vector" nodes there are other transformation nodes, which allow for the transformation of tags to strings, terms to strings and vice versa. The "Term to Structure" node converts terms that have been recognized as chemical compounds into,(e.g. SIMLES structures), which can then be rendered 2D later on. The "BoW creator" node, which transforms a list of documents into a bag of words is available in the Transformation node category as well. A very useful transformation node is the "Strings to Documents" node. This node requires an input data table consisting of several string columns. For each row a document is created. Each of the contained columns is used as a certain text field in the created documents. It can be specified which columns contain the abstracts, the full texts or the titles of the documents to create. This node comes in handy when the textual data is available only in, for example, csv format.

Visualization

There are two nodes that visualize textual data, the "Document Viewer" and the "Tag Cloud" node. The "Document Viewer" node allows for a simple visualization of documents. A data table consisting of at least one column containing *DocumentCells* is required as input data table. The view of the "Document Viewer" node shows a table of all document titles of the input data table, as illustrated in Figure 6.

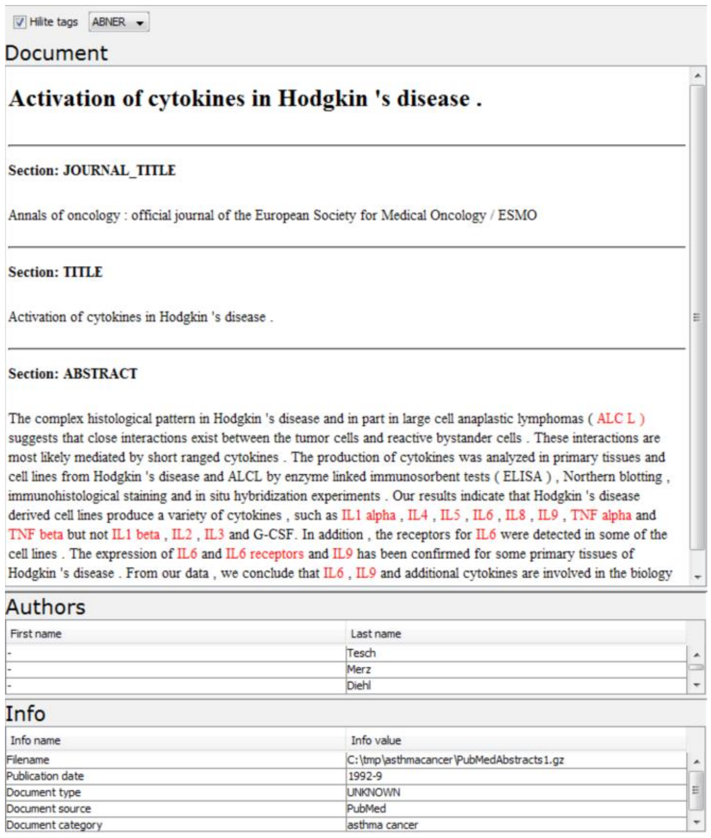


Documents

Document title
A common haplotype of interleukin-4 gene IL4 is associated with severe respiratory syncytial virus disease in Korean children .
A human T-cell line with inducible production of interleukins 5 and 4 . A model for studies of gene expression .
A novel cytokine pathway suppresses glial cell melanogenesis after injury to adult nerve .
A novel cytokine-inducible gene CIS encodes an SH2-containing protein that binds to tyrosine-phosphorylated interleukin 3 ...
A strategy for treatment of Epstein-Barr virus-positive Hodgkin 's disease by targeting interleukin 12 to the tumor environm...
Aberrant responses of human lymphocytic neoplasms to cytokine regulation .
Activation of cytokines in Hodgkin 's disease .
Activation-induced cytidine deaminase (AID)-associated multigene signature to assess impact of AID in etiology of diseases ...
Adult asthmatics display exaggerated IFNgamma responses to human metapneumovirus and respiratory syncytial virus .
Alteration of cellular and humoral immunity by mutant p53 protein and processed mutant peptide in head and neck cancer .
Analysis of T-cell proliferation and cytokine secretion in the individuals exposed to arsenic .
Association of cord blood cytokine levels with wheezy infants in the first year of life .
Beneficial health effects of lupeol triterpene : a review of preclinical studies .
Characterization and growth factor requirements of SJL lymphomas . II . Interleukin 5 dependence of the in vitro cell line , c...
Clinical impact of different classes of infiltrating T cytotoxic and helper cells (Th1 , th2 , treg , th17) in patients with colorec...
Common genetic variants and risk for non-Hodgkin lymphoma and adult T-cell lymphoma/leukemia in Jamaica .
Comparison of allergic lung disease in three mouse strains after systemic or mucosal sensitization with ovalbumin antigen .
Comparison of salmeterol with placebo in mild asthma : effect on peripheral blood phagocyte function and cytokine levels .
Conserved responses to trichostatin A in rodent lungs exposed to endotoxin or stretch .
Contribution of bronchial fibroblasts to the antiviral response in asthma .

Figure 6: A table showing the titles of all documents of the input data table of the "Document Viewer" node.

A double click on a particular row (document title) will open a new window showing the details of the corresponding document. All available data of a document is shown including the meta data. In the bottom most part of the panel, the documents' filename, publication date, document type, categories and sources are displayed. In the panel above, all author names are shown and in the main panel the text of the document and its title are displayed. Furthermore, assigned tags of a specified type can be highlighted, which can be seen in Figure 7. Terms that have been recognized as biomedical named entities (gene and protein names) by the “Abner tagger” node are emphasized in red.



Document

Activation of cytokines in Hodgkin 's disease .

Section: JOURNAL_TITLE

Annals of oncology : official journal of the European Society for Medical Oncology / ESMO

Section: TITLE

Activation of cytokines in Hodgkin 's disease .

Section: ABSTRACT

The complex histological pattern in Hodgkin 's disease and in part in large cell anaplastic lymphomas (ALC L) suggests that close interactions exist between the tumor cells and reactive bystander cells . These interactions are most likely mediated by short ranged cytokines . The production of cytokines was analyzed in primary tissues and cell lines from Hodgkin 's disease and ALCL by enzyme linked immunosorbent tests (ELISA) , Northern blotting , immunohistological staining and in situ hybridization experiments . Our results indicate that Hodgkin 's disease derived cell lines produce a variety of cytokines , such as IL1 alpha , IL4 , IL5 , IL6 , IL8 , IL9 , TNF alpha and TNF beta but not IL1 beta , IL2 , IL3 and G-CSF. In addition , the receptors for IL6 were detected in some of the cell lines . The expression of IL6 and IL6 receptors and IL9 has been confirmed for some primary tissues of Hodgkin 's disease . From our data , we conclude that IL6 , IL9 and additional cytokines are involved in the biology

Authors

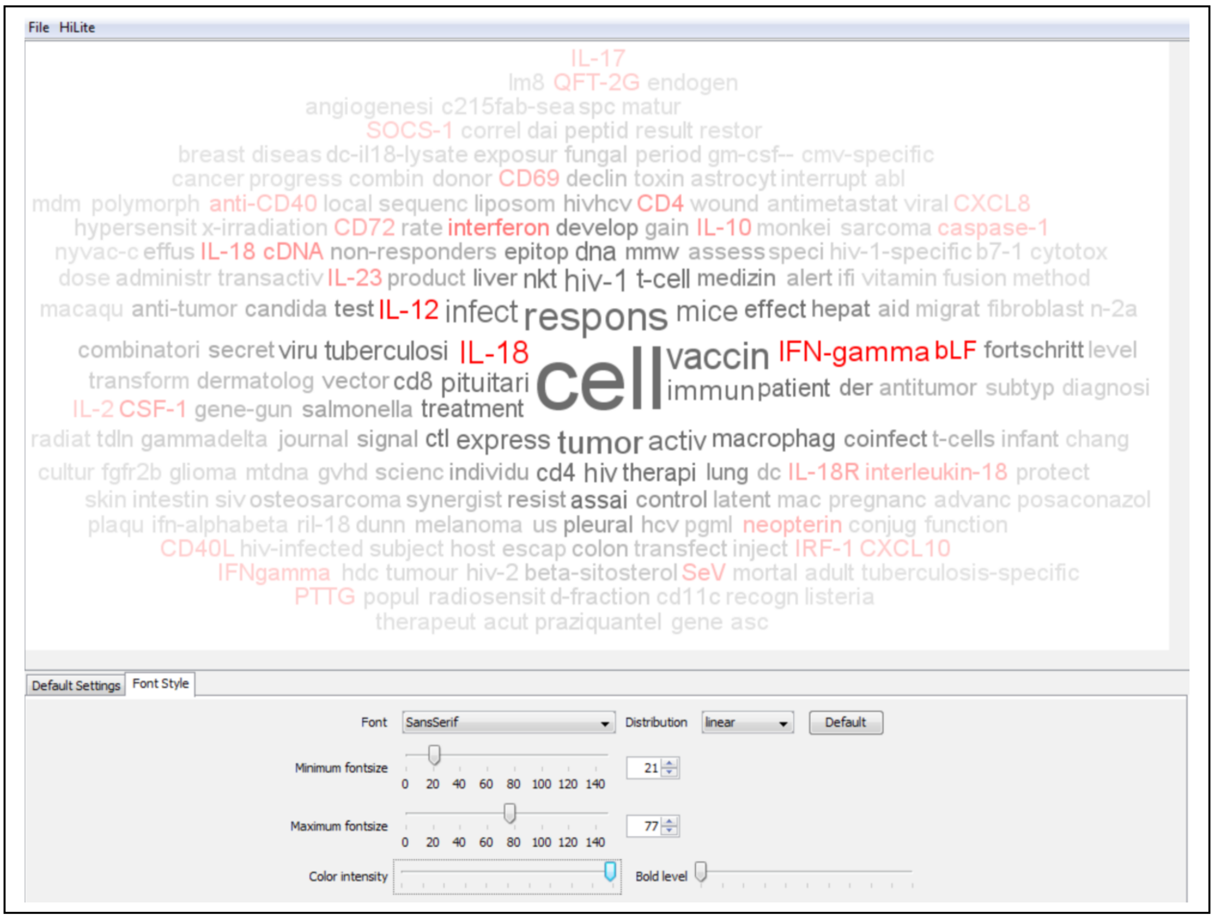
First name	Last name
-	Tesch
-	Merz
-	Diehl

Info

Info name	Info value
Filename	C:\tmp\asthmacancer\PubMedAbstracts1.gz
Publication date	1992-9
Document type	UNKNOWN
Document source	PubMed
Document category	asthma cancer

Figure 7: The detailed view of a particular document, showing all its data, such as authors, categories, publication date, text etc., as well as the recognized gene and protein names are emphasized in red.

The “Tag Cloud” node creates a typical tag cloud with additional options such as different arrangements of terms, (i.e. size sorted, alphabetical, inside out, adjustable minimal and maximal font size, or transparency of term)s. The node requires an input data table consisting of a bag of words with an additional column containing a frequency, weight, or score of each term. This weight is used to determine the font size of the term, its transparency and boldness. The “Tag Cloud” node provides an image out port, meaning that the created tag cloud can be used (e.g. in KNIME reports). Figure 8 shows a tag cloud containing the 500 terms with the highest term frequency.


File HiLite
IL-17
lm8 QFT-2G endogen
angiogenesi c215fab-sea spc matur
SOCS-1 correl dai peptid result restor
breast diseas dc-il18-lysate exposur fungal period gm-csf-- cmv-specific
cancer progress combin donor CD69 declin toxin astrocyt interrupt abl
mdm polymorph anti-CD40 local sequenc liposom hivhcv CD4 wound antimetastat viral CXCL8
hypersensit x-irradiation CD72 rate interferon develop gain IL-10 monkei sarcoma caspase-1
nyvac-c effus IL-18 cDNA non-responders epitop dna mmw assess speci hiv-1-specific b7-1 cytotox
dose administr transactiv IL-23 product liver nkt hiv-1 t-cell medizin alert ifi vitamin fusion method
macaqu anti-tumor candida test IL-12 infect respons mice effect hepat aid migrat fibroblast n-2a
combinatori secret viru tuberculosi IL-18 cell vaccin IFN-gamma bLF fortschritt level
transform dermatolog vector cd8 pituitari immun patient der antitumor subtyp diagnosi
IL-2 CSF-1 gene-gun salmonella treatment
radiat tdln gammadelta journal signal ctl express tumor activ macrophag coinfect t-cells infant chang
cultur fgfr2b glioma mtdna gvhd scienc individu cd4 hiv therapi lung dc IL-18R interleukin-18 protect
skin intestin siv osteosarcoma synergist resist assai control latent mac pregnanc advanc posaconazol
plaqu ifn-alphabeta ril-18 dunn melanoma us pleural hcv pgml neopterin conjug function
CD40L hiv-infected subject host escap colon transfect inject IRF-1 CXCL10
IFNgamma hdc tumour hiv-2 beta-sitosterol SeV mortal adult tuberculosis-specific
PTTG popul radiosensit d-fraction cd11c recogn listeria
therapeut acut praziquantel gene asc
Default Settings
Font Style
Default
0 20 40 60 80 100 120 140
Color intensity
Bold level

Font: SansSerif
Distribution: linear
Minimum fontsize: 21
Maximum fontsize: 77

Figure 8: A tag cloud view.

It can be seen that named entities recognized and tagged by a tagger node of the enrichment category can be colored differently. In Figure 8 the recognized gene and protein names are emphasized in red. Since the “Tag Cloud” node supports coloring that has been assigned beforehand by the “Color Manager” node the color handling is very flexible. More examples of tag cloud visualizations can be seen at http://tech.knime.org/gene-and-protein-tag-cloud-example or http://tech.knime.org/named-entity-recognizer-and-tag-cloud-example.

Data Structures

In this section the different structures of data tables required by the nodes of the Text Processing feature, as well as the new data types, *Document(Blob)Cell* and *TermCell* are described.

Data Table Structures

As already mentioned in Section "Philosophy", certain nodes require a certain structure of the input data table in order to configure and execute properly. What kind of data table structure is specified by the category of the nodes. There are three different kind of structures:

- A data table with only one column containing *DocumentCells*, shown in Figure 2.
- A bag of words data table consisting of two columns, one containing *DocumentCells*, the other containing *TermCells*, illustrated in Figure 4. Furthermore additional columns containing numerical data are possible in a bag of words, (e.g. columns that contain different frequencies).

- A data table containing term or document vectors, shown in Figure 11.

A single row of a bag of words data table represents the occurrence of a particular term (of the term column) in a certain document (of the document column). This means that for each term contained in a document, one row in the corresponding bag of words data table is created. The basic bag of words data table can be extended by numerical columns that contain frequencies or the original unchanged documents, which is useful when deep preprocessing is activated. Figure 9 shows such kind of a bag of words data table where the original document is appended in an additional column.

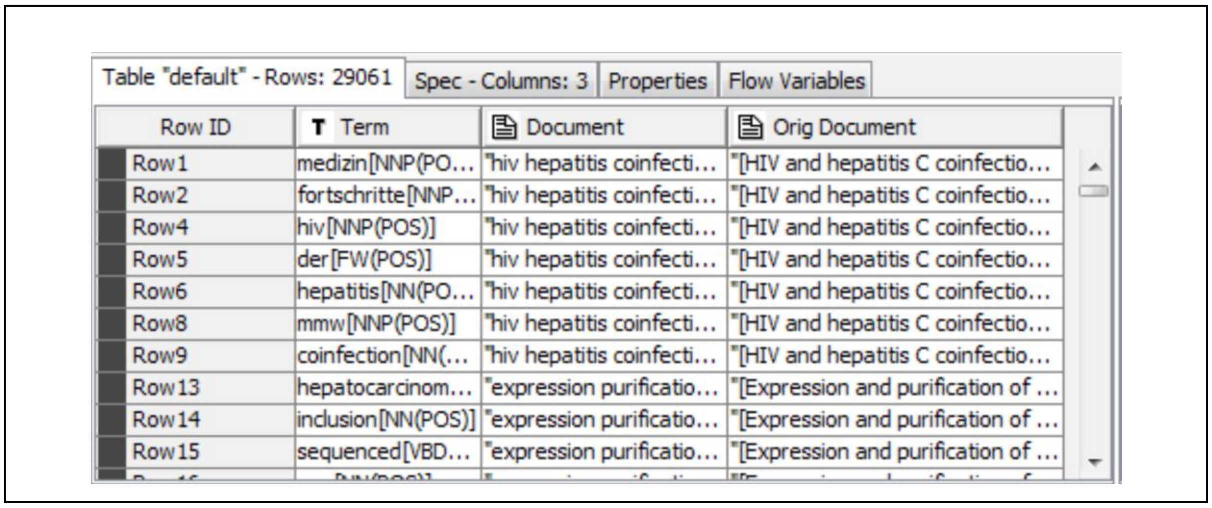


Table "default" - Rows: 29061 | Spec - Columns: 3 | Properties | Flow Variables

Row ID	Term	Document	Orig Document
Row1	medizin[NNP(PO…	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row2	fortschritte[NNP…	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row4	hiv[NNP(POS)]	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row5	der[FW(POS)]	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row6	hepatitis[NN(PO…	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row8	mmw[NNP(POS)]	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row9	coinfection[NN(…	"hiv hepatitis coinfecti…	"[HIV and hepatitis C coinfectio…
Row13	hepatocarcinom…	"expression purificatio…	"[Expression and purification of …
Row14	inclusion[NN(POS)]	"expression purificatio…	"[Expression and purification of …
Row15	sequenced[VBD…	"expression purificatio…	"[Expression and purification of …

Figure 9: A bag of words data table with an additional column containing the original, unchanged document.

It can be seen that the document titles of the column in the middle and those of the right column differ. The original documents on the right still have the original titles, whereas the titles of the documents in the middle column, affected by deep preprocessing, have been changed by preprocessing nodes, (e.g. stop word filtering). Not only the body of the documents will be changed by preprocessing, the titles will be changed as well.

Figure 10 illustrates a bag of words data table with additional columns containing frequencies of terms, such as the absolute and relative term frequency and the inverse document frequency.

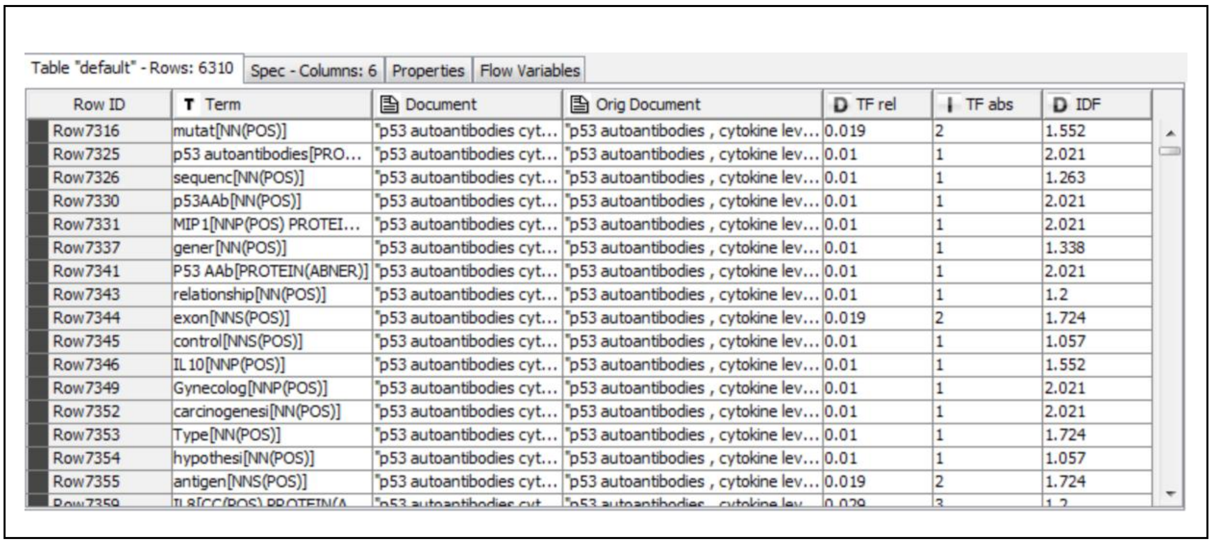


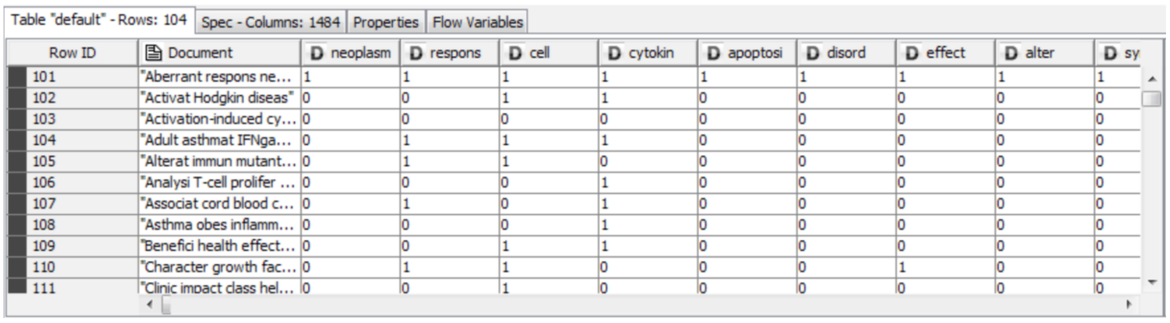
Table "default" - Rows: 6310 | Spec - Columns: 6 | Properties | Flow Variables

Row ID	Term	Document	Orig Document	TF rel	TF abs	IDF
Row7316	mutat[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.019	2	1.552
Row7325	p53 autoantibodies[PRO…	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7326	sequenc[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.263
Row7330	p53AAb[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7331	MIP1[NNP(POS) PROTEI…	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7337	gener[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.338
Row7341	P53 AAb[PROTEIN(ABNER)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7343	relationship[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.2
Row7344	exon[NNS(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.019	2	1.724
Row7345	control[NNS(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.057
Row7346	IL10[NNP(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.552
Row7349	Gynecolog[NNP(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7352	carcinogenesi[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	2.021
Row7353	Type[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.724
Row7354	hypothesi[NN(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.01	1	1.057
Row7355	antigen[NNS(POS)]	"p53 autoantibodies cyt…	"p53 autoantibodies , cytokine lev…	0.019	2	1.724

Figure 10: A bag of words data table with additional columns containing different term frequencies.

The numerical (or binary) representation of documents or terms is a document or term vector. With the "Document vector" or "Term vector" node it is possible to create numerical or bit vectors, one for each document or term. A document vector represents a document in the term space of the corpus and a term vector represents a term in the document space of the corpus. Whether binary or numerical vectors will be created, can be specified in the dialog of the corresponding node. Any frequency that is contained in the bag of words can be used as numerical value to create the vectors.

Figure 11 illustrates a binary document vector data table. The first column contains the documents, the following columns, one for each term of the corpus, contain either a 1 if the corresponding term is contained in the document or a 0 otherwise. The header of each column shows the corresponding term. Based on these vectors, regular KNIME nodes can be applied to cluster or classify documents or terms.

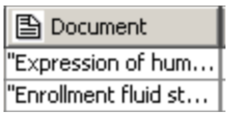


Row ID	Document	neoplasm	respons	cell	cytokin	apoptosi	disord	effect	alter	sy
101	"Aberrant respons ne...	1	1	1	1	1	1	1	1	1
102	"Activat Hodgkin diseas"	0	0	1	1	0	0	0	0	0
103	"Activation-induced cy...	0	0	0	0	0	0	0	0	0
104	"Adult asthmat IFNga...	0	1	1	1	0	0	0	0	0
105	"Alterat immun mutant...	0	1	1	0	0	0	0	0	0
106	"Analysi T-cell prolifer ...	0	0	0	1	0	0	0	0	0
107	"Associat cord blood c...	0	1	0	1	0	0	0	0	0
108	"Asthma obes inflamm...	0	0	0	1	0	0	0	0	0
109	"Benefici health effect...	0	0	1	1	0	0	0	0	0
110	"Character growth fac...	0	1	1	0	0	0	1	0	0
111	"Clinic impact class hel...	0	0	1	0	0	0	0	0	0

Figure 11: A data table consisting of binary document vectors.

Data Types

Almost all of the input and output data tables of the text processing nodes contain documents and/or terms. To encapsulate documents and terms, two new data types have been designed and implemented; the *Document(Blob)Cell* and the *TermCell*.



Document
"Expression of hum...
"Enrollment fluid st...

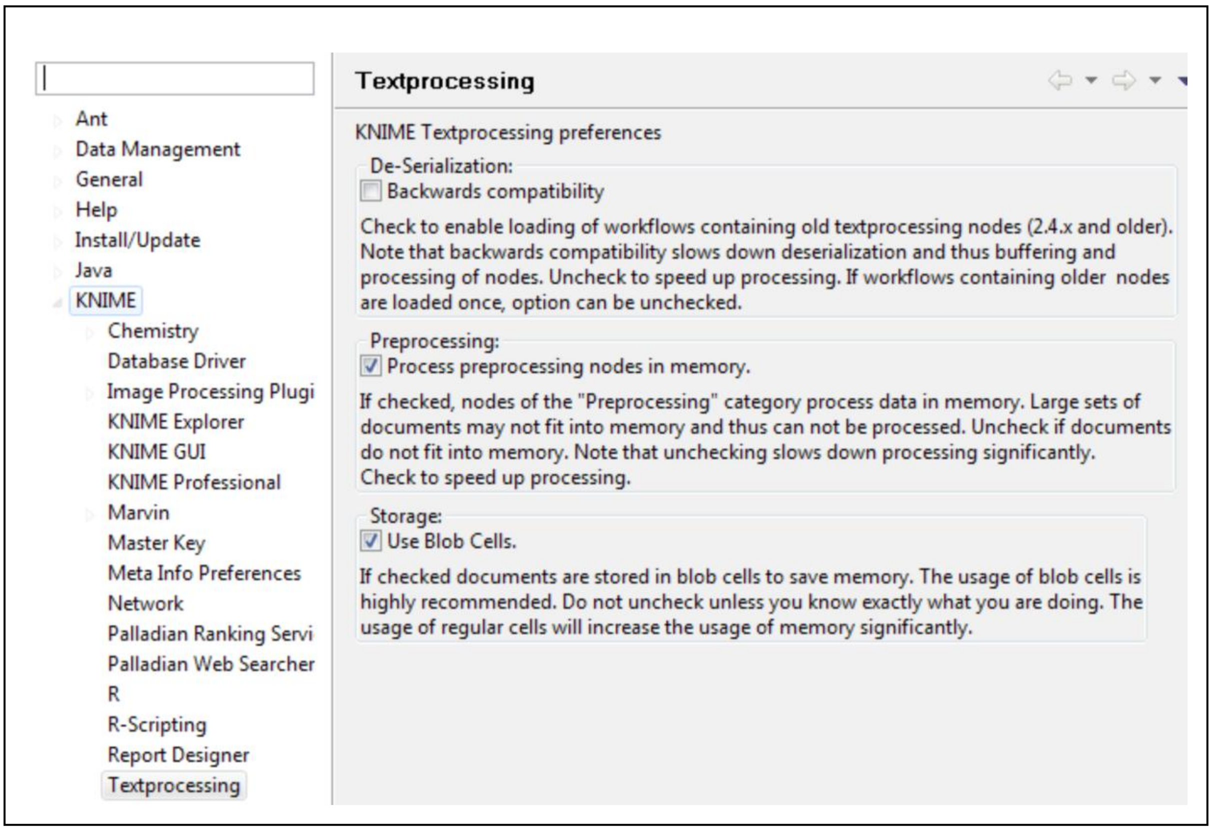
Figure 12: A column containing two Document(Blob)Cells.

Document(Blob)Cell

Cells containing documents are illustrated in Figure 12. The icon, shown at the header of the column, symbolizes that the column contains *DocumentBlobCells* or *DocumentCells*. Each cell shows only the title of the contained document.

Both, the *DocumentBlobCells* and the *DocumentCells* encapsulate documents. The only difference is that the *DocumentCells* are not KNIME *BlobDataCells*. The advantage of the *DocumentBlobCell* in comparison to the *DocumentCell* is that multiple occurrences of a certain cell in a data table are referenced. The cells are not duplicated during the creation of the data table. Especially during the creation of bag of words data tables, where for each term occurring in a document a row, containing the term and the document is created, the usage of *DocumentBlobCells* can save a lot of memory. *DocumentBlobCells* are used by default by all nodes of the Text Processing feature.

The usage of blob cells as well as other global option can be specified in the preference page of the Text Processing feature, as illustrated in Figure 13.



Textprocessing

KNIME Textprocessing preferences

De-Serialization:

☐ Backwards compatibility

Check to enable loading of workflows containing old textprocessing nodes (2.4.x and older). Note that backwards compatibility slows down deserialization and thus buffering and processing of nodes. Uncheck to speed up processing. If workflows containing older nodes are loaded once, option can be unchecked.

Preprocessing:

☑ Process preprocessing nodes in memory.

If checked, nodes of the "Preprocessing" category process data in memory. Large sets of documents may not fit into memory and thus can not be processed. Uncheck if documents do not fit into memory. Note that unchecking slows down processing significantly. Check to speed up processing.

Storage:

☑ Use Blob Cells.

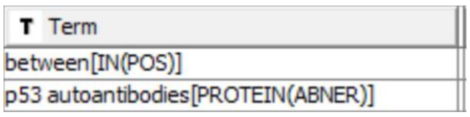
If checked documents are stored in blob cells to save memory. The usage of blob cells is highly recommended. Do not uncheck unless you know exactly what you are doing. The usage of regular cells will increase the usage of memory significantly.

Figure 13: Preference page of the Text Processing feature.

The advantage of not using blob cells is an increase of processing speed since the references don't have to be resolved. However, it is strongly recommended not to switch off the usage of blob cells.

TermCell

For the representation of terms, the *TermCell* is provided by the feature. A term can contain one or more words, (e.g. the multi word “text mining”, or the gene name “interleukin 6'”) can represent a term. In addition, a term can contain one or multiple tags assigned by enrichment nodes. After parsing of a document, each single word represents a term with no tags assigned. The enrichment nodes used after the parser nodes bundle words if they have been recognized as a named entity and a tag is assigned to that particular term. Figure 14 illustrates a column containing *TermCells*. In the column heading the term cell icon is displayed. The string representation of a term, shown in table views is assembled by its words first followed by the attached tags in brackets. First the name of the tag is shown, (e.g. “PROTEIN'” for a protein name) and second the type of the tag,(e.g. “ABNER”) indicating the tagger node which assigned the tag.



T Term
between[IN(POS)]
p53 autoantibodies[PROTEIN(ABNER)]

Figure 14: A column containing two TermCells.

Conclusion

In this paper the KNIME Text Processing feature has been introduced. The functionality as well as the philosophy and usage of the feature has been described. It has been shown which nodes need to be applied in which order to parse documents, add semantic information by named entity recognition, preprocess the documents, and finally visualize them or create numerical vectors which can be used by regular KNIME nodes for clustering or classification. Additionally two new data types, provided by the feature, the *DocumentCell* and the *TermCell* have been described, encapsulating documents or terms respectively. More information and example workflows for clustering and classification of documents, usage of tag clouds and other visualizations can be found at http://tech.knime.org/examples.

Acknowledgments

Thanks to Pierre-Francois Laquerre for his work on the on “Chi-square keyword extractor” node and “Keygraph keyword extractor” node as well as to Iris Adä for her work on the “Tag Cloud” node.

References

1: Berthold, M.R., Cebron, N., Dill, F., Gabriel, T.R., Kötter, T., Meinl, T., Ohl, P., Thiel, K., Wiswedel, B.: KNIME - The Konstanz Information Miner. SIGKDD Explorations11(1) (2009)
2: Settles, B.: Abner: an open source tool for automatically tagging genes, proteins and other entity names in text. Bioinformatics21(14) (2005) 3191-3192
3: Matsuo, Y., Ishizuka, M.: Keyword extraction from a single document using word co-occurrence statistical information. International Journal on Artificial IntelligenceTools13(1) (2004) 157-169
4: Ohsawa, Y., Benson, N.E., Yachida, M.: KeyGraph: automatic indexing by co-occurrence graph based on building construction metaphor. IEEE International Forum on Research and Technology Advances in Digital Libraries, Proceedings (1998) 12-18