Generating SD-Rules in the SPECIALIST Lexical Tools
Optimization for Suffix Derivation Rule Set

Chris J. Lu[1,2], Destinee Tormey[1], Lynn McCreedy[1] and Allen C. Browne[1]
[1]*National Library of Medicine, Bethesda, MD, U.S.A.*
[2]*Medical Science & Computing, LLC, Rockville, MD, U.S.A.*

Keywords: Derivations, Suffix Derivations, SD-Rules, Natural Language Processing, the SPECIALIST Lexical Tools.

Abstract: Suffix derivations (SDs) are used with query expansion in concept mapping as an effective Natural Language Processing (NLP) technique to improve recall without sacrificing precision. A systematic approach was proposed to generate derivations in the SPECIALIST Lexical Tools in which SD candidate rules were used to retrieve SD-pairs from the SPECIALIST Lexicon (Lu et al., 2012). Good SD candidate rules are gathered as SD-Rules in Lexical Tools for generating SDs that are not known to the Lexicon. This paper describes a methodology to select an optimized SD-Rule set that meets our requirement of 95% system precision with best system performance from SD candidate rules. The results of the latest three releases of Lexical Tools show: 1) system precision and recall of selected SD-Rules are above 95%. 2) a consistency between a computational linguistic approach and traditional linguistic knowledge for selecting the best Parent-Child rules. 3) a consistent approach yielding similar SD-Rule sets and system performance. Ultimately, it results in better precision and recall for NLP applications using Lexical Tools derivational related flow components.

1 INTRODUCTION

NLP in medicine is used to retrieve information and analyze linguistic patterns from electronic medical records (EMRs), published biomedical research, clinical trial reports, and other sources. Due to the great deal of lexical variation in natural language, managing this variability is an important key to successful Medical Language Processing (Pacak et al., 1980; Wolff, 1987). The National Library of Medicine (NLM) distributes the NLP SPECIALIST Lexicon and Lexical Tools as one of the Unified Medical Language System (UMLS) Knowledge Sources along with the Metathesaurus. These rich and robust NLP resources have provided the NLP/Medical Language Processing community with an extensive NLP toolset since 1994 (McCray et al., 1993). The SPECIALIST Lexical Tools is designed to handle lexical variation, including derivational variant generation.

Derivational variants are words related by a derivational process, such as suffixation, pre-fixation, and conversion, to create new words based on existing words. They need not be synonymous. In fact, derivation often entails thoroughgoing meaning change. Derivations allow users to find closely related terms that may differ in syntactic category, but are nonetheless usefully related. They are used with query expansion to increase recall for UMLS concept mapping in MetaMap (Aronson and Lang, 2010) and Sophia (Divita et al., 2014). For example, no CUI was found by direct mapping if the source vocabulary is "perforated ear drum". By substituting the subterm "perforated" for its derivational variant, "perforation," the UMLS Metathesaurus concept, C0206504, is found. More information, such as its preferred term (Tympanic Membrane Perforation) and synonyms, can be retrieved for further NLP analysis.

SD generation in the Lexical Tools derivational flow component is based on a paradigm of SD-Facts and SD-Rules (Lu et al., 2012). Over 100 SD candidate rules have been collected by the Lexical Systems Group (LSG). All terms in the Lexicon that match a SD candidate rule are retrieved as SD candidate pairs. About 30% of them are validated automatically by a computer expert system while the rest (~70%) are tagged by LSG linguists manually (Lu et al., 2013). Valid SD-pairs are stored in the Lexical Tools database to retrieve derivations that are known to the Lexicon, that is, SD-Facts. The increase of both the Lexicon and SD candidate rules

Lu, C., Tormey, D., McCreedy, L. and Browne, A.
Generating SD-Rules in the SPECIALIST Lexical Tools - Optimization for Suffix Derivation Rule Set.
DOI: 10.5220/0005731303530358
In *Proceedings of the 9th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2016) - Volume 5: HEALTHINF*, pages 353-358
ISBN: 978-989-758-170-0

contributes to the growth of SD-Facts. The number of SD-Facts has increased over 12.4 times from 4,110 to 50,814 since 2011 for better recall. The derivational flow component's algorithm is enhanced by this basis on SD-Facts, while it was mainly based on SD-Rules in 2011, for better precision. As a result, the Lexicon and Lexical Tools provide one of the most comprehensive and robust resources of derivations for the NLP community.

If no derivations are found by SD-Facts, SD-Rules are used to generate SDs in Lexical Tools. SD-Rules are good rules selected from the SD candidate rules and stored in a Trie mechanism (Aho et al., 1983). Four heuristic algorithms are implemented in Lexical Tools to eliminate nonrealistic derivational variants and increase precision (Lu et al., 2012).

SD candidate rules can be derived by computer algorithms from a non-derivational lexicon (Gaussier, 1999), based on synonym relations (Grabar and Zweigenbaum, 1999), structured terminology (Grabar and Zweigenbaum, 2000), etc. Derivations from Lexical Tools were used as a gold standard for comparison in the above studies. There are hundreds of SD candidate rules. Due to limited resources, the LSG evaluated the most common English suffixes for SD candidate rules. Before these candidate rules can become SD-Rules in the Lexical Tools, they undergo evaluation to ensure they provide the required system performance. This paper describes a methodology of: 1) finding the SD-Rules - the optimized subset from collected SD candidate rules, and 2) finding the best SD rule(s) in a Parent-Child (P-C) family.

2 OBJECTIVE AND APPROACH

Our goal is to find a set of SD-Rules for generating derivational variants that are not unknown to the Lexicon using Lexical Tools. This set of SD-Rules is a subset of SD candidate rules selected by filtering out bad rules. This subset must meet our objectives of cumulative system precision above 95% and best system performance (see section 2.3). In addition, rules in a P-C family with the best system performance are chosen for the optimal set. The methodology is described below.

2.1 New SD Candidate Rules

The LSG has enhanced SD generation in the Lexicon and Lexical Tools by adding new SD candidate rules and Lexical records. There are 13 and 11 new SD candidate rules introduced in the 2015 and 2016 releases, as shown in Tables 1 and 2. A SD rule includes suffixes and parts of speech for a candidate SD pair. For example, the suffix "es" in certain nouns can be replaced with the suffix "ic" to create a related adjective (adj). Thus, replacing "es" with "ic" in "diabetes" creates "diabetic", expressible as SD-pair diabetes|noun|diabetic|adj. This SD rule (ES-15-03 in Table 1) is coded in the following format in Lexical Tools: es$|noun|ic$|adj, where "$" means the end of the word. SD rules can be applied to generate SD-pairs in both directions; "diabetic" generates "diabetes" from this same rule. Invalid SD-pairs (exceptions to this SD candidate rule) such as, comes|noun|comic|adj, are also retrieved from Lexicon. These candidate SD-pairs are validated by computer programs and linguists during the derivation generation process (Lu et al., 2012).

Table 1: New SD Candidate Rules in Lexicon, 2015.

ID	New SD Candi. Rule	Rank
CG-15-01	se$\|verb\|zation$|noun	2
CG-15-02	sation$\|noun\|ze$|verb	3
CG-15-03	ility$\|noun\|le$|adj	9
CG-15-04	$\|adj\|ally$|adv	15
CG-15-05	ce$\|noun\|t$|adj	18
CG-15-06	cy$\|noun\|t$|adj	19
CG-15-07	e$\|verb\|ion$|noun	20
CG-15-08	c$\|adj\|s$|noun	43
ES-15-01	e$\|verb\|ing$|noun	45
ES-15-02	al$\|adj\|us$|noun	61
ES-15-03	es$\|noun\|ic$|adj	67
ES-15-04	$\|noun\|ize$|verb	78
ES-15-05	es$\|noun\|ic$|noun	101

Table 2: New SD Candidate Rules in Lexicon, 2016.

ID	New SD Candi. Rule	Rank
CG-16-01	se$\|verb\|sis$|noun	27
CG-16-02	sia$\|noun\|tic$|adj	40
CG-16-03	on$\|noun\|ve$|adj	48
CG-16-04	e$\|noun\|ic$|adj	49
CG-16-05	$\|adj\|ism$|noun	51
CG-16-06	ation$\|noun\|ed$|adj	67
CG-16-07	$\|noun\|ship$|noun	70
CG-16-08	e$\|adj\|ion$|noun	88
CG-16-09	$\|noun\|age$|noun	96
ES-16-01	esis$\|noun\|ic$|adj	13
ES-16-02	al$\|adj\|ine$|noun	98

New rules suggested by NLP experts, including Lexicon and Lexical Tools users and LSG linguists, are marked as ES# (expert-suggested). In addition to ES candidate rules, the LSG developed a system to derive SD candidates from known SD-pairs automatically. First, the LSG collected all known SD-pairs from nominalizations (nomD) in the Lexicon

and the original SD-pairs (orgD) that existed in the deprecated Lexical Tools release of 2011. These orgDs are not necessary in the current SD-Facts, because some of them were removed after the new derivation generation system was deployed in 2012 (Lu et al., 2012). Next, computer programs found SD candidate rules by removing all common leading characters (the word stem) of the SD-pairs. For example, nomD, celebrate|verb|celebration|noun, is collected from the Lexical record celebrate|E0015729, as shown in Figure 1. The SD rule, e$|verb|ion$|noun, is derived by removing the stem "celebrat" from "celebrate" and "celebration". In the 2015 release, 1,017 SD rules were derived from 23,384 nomDs and sorted by frequency. The top 17 rules with the highest frequency cover above 80% of all nomDs. Six new rules (non-existent in the previous SD rule set) were selected. Similarly, 1,421 SD rules were derived from 4,110 orgDs. Two new rules from the top 6 with the highest frequency were selected. These 8 (6 from nomDs and 2 from orgDs) new computer-generated rules were added to the 2015 SD candidate rules, marked as CG# in Table 1. For

```
{base=celebrate
entry=E0015729
    cat=verb
    variants=reg
    intran
    tran=np
    nominalization=celebration|noun|E0015730
}
```

Figure 1: Lexical Record of "celebrate".

the 2016 release, 9 computer-generated SD rules (4 from nomDs and 5 from orgDs) from the following top ranking have similarly been selected, as shown in Table 2.

2.2 Establish the Baseline

We will use 2015 as an example to illustrate these processes. First, the 13 new SD rules (5 expert-suggested and 8 computer-generated) are added to the 107 SD candidate rules of the prior release (2014). Second, these rules are normalized and ordered alphabetically. The result is 120 unique SD candidate rules in a standard format. Third, all rules related in a P-C family in the new SD candidate set are identified. Table 3 identifies 14 Parent-Rules (P#) and 19 associated Child-Rules (C#) for 2015 release. Fourth, all 19 Child-Rules are removed to avoid duplication, because the best rules from all generations for all these 14 P-C families will be chosen through the optimization processes (see section 2.4). This set of normalized SD candidate rules, composed of 101 (= 120 – 19) unique rules (with no Child-Rules), is used as the baseline for further processes.

2.3 System Performance

Our goal is to choose a set of SD-Rules that perform as well as SD-Facts in Lexical Tools. Retrieved candidate SD-pairs (retrieved instances) and valid SD-pairs in SD-Facts (relevant instances) are used to calculate precision and recall. First, we retrieve all

Table 3: Parent-Child SD Rule Families, 2015.

PID	Parent-Rule	CID	Child-Rule
P1	$\|adj\|ally$|adv	C1	ic$\|adj\|ically$|adv
P2	$\|adj\|ity$|noun	C2	ic$\|adj\|icity$|noun
P3	$\|noun\|al$|adj	C3	ic$\|noun\|ical$|adj
P4	$\|verb\|ion$|noun	C4	ss$\|verb\|ssion$|noun
P5	a$\|noun\|an$|adj	C5	ia$\|noun\|ian$|adj
P6	a$\|noun\|an$|noun	C6	ia$\|noun\|ian$|noun
P7	a$\|noun\|ar$|adj	C7	ula$\|noun\|ular$|adj
P8	ation$\|noun\|e$|verb	C8	ization$\|noun\|ize$|verb
P9	c$\|adj\|s$|noun	C9	ic$\|adj\|is$|noun
P10	ce$\|noun\|t$|adj	C10 C11 C12	ance$\|noun\|ant$|adj iance$\|noun\|iant$|adj ence$\|noun\|ent$|adj
P11	cy$\|noun\|t$|adj	C13 C14	ency$\|noun\|ent$|adj iency$\|noun\|ient$|adj
P12	e$\|verb\|ion$|noun	C15 C16	ate$\|verb\|ation$|noun se$\|verb\|sion$|noun
P13	ility$\|noun\|le$|adj	C17	ability$\|noun\|able$|adj
P14	sis$\|noun\|tic$|adj	C18 C19	esis$\|noun\|etic$|adj osis$\|noun\|otic$|adj

SD-pairs that match SD rules in the evaluation set. Second, we sort these unique SD candidate rules by rank. That is, by the descending order of precision (= relevant, retrieved No./retrieved No.), frequency (retrieved No.), and then the alphabetical order of SD rules. Third, we calculate the cumulative system precision (SP) and system recall (SR = relevant, retrieved No./relevant No.) for all rules. Fourth, we find the cutoff rule with the least SP above 95%. All SD candidate rules with a higher rank than the cutoff rule are selected as good rules for the SD-Rule set. The system performance of the SD-Rule set is measured by both (sum of) cumulative SP and SR at the cutoff rule.

2.4 Parent-Child Rules Optimization

In Table 3, C9, ic$|adj|is$|noun, is the first generation Child-Rule of P9 by adding one character, 'i', to P9, c$|adj|s$|noun. P9 is a root Parent-Rule because it does not have a possible Parent-Rule (no common leading characters). Note that all computer-generated SD candidates are root Parent-Rules. Parent-Rules might have multiple Child-Rules, such as Parent-Rule P14, which has two Child-Rules, C18 and C19, by adding "e" and "o". Further, multiple generations of Child-Rules could be involved. For example, the root Parent-Rule, P10, has 2 second generation Child-Rules C10 and C12 (by adding the two characters "an" and "en", respectively) and 1 third generation Child-Rule C11 (by adding the three characters "ian"). In such a case, C10 is both a Child-Rule (of P10) and a Parent-Rule (of C11).

The root Parent-Rule is used to find the best rule(s) in its P-C family. Theoretically, a Parent-Rule should have better recall than its Child-Rule because SD-pairs derived from a Child-Rule are a subset of those derived from the associated Parent-Rule. A Parent-Rule, however, could have worse precision than its Child-Rule because it could include more invalid SD-pairs. Local precision (valid SD-pairs over retrieved SD candidate pairs from the rule and Lexicon) and frequency (instances of the retrieved SD candidate pairs from the rule and Lexicon) are two criteria we used for finding the qualified Child-Rules in a P-C family for further evaluation. A sophisticated heuristic algorithm has been developed. First, the root Parent-Rule of the P-C family replaces the original Parent-Rules (P1 – P14). For example, the root Parent-Rule, $|noun|n$|adj, replaces P5 (by removing character 'a' from P5). Second, all possible Child-Rules from all generations are derived from this root Parent-Rule. Third, qualified Child-Rules are selected from all generations if they 1) have higher precision than their Parent-Rules; and 2) meet frequency criteria (> 25% of their root Parent-Rule and > 40% of their immediate Parent-Rule). Finally, the system performance of all qualified P-C rules of a P-C family are calculated. The one with the best system performance is chosen. Linguistic knowledge guides the P-C choice when the system performance of different generations of P-C rules are the same. Finally, a Parent-Rule is chosen over a Child-Rule for better recall when the above conditions are equal. This process is repeated for finding the best P-C rules from all 14 P-C families to obtain the optimized SD-Rule set.

3 RESULTS AND CONCLUSION

Table 4 summarizes the results of generating the optimized SD-Rule set from 2014-2016. The system performance (~1.90), SP (~95%) and SR (~95%) of the SD-Rule sets stay consistent as the number of candidate SD rules and retrieved SD-pairs increases about 10% each year. Figure 2 shows a similar pattern of the system precision-recall vs. SD candidate rules in the optimized rule set for 2014-2016. The consistency over three releases indicates this approach is stable. The intersections of precision and recall curves are at the cutoff SD rule with precision and recall both around 95%. This equilibrium between precision and recall confirms that setting the precision at 95% as the minimum cutoff requirement is the right choice for representing system performance.

The 2014 and 2015 releases have the same cutoff SD rule, ar$|adj|e$|noun. This rule drops to rank 83 (just below the cutoff rank of 82) in 2016. Rules above the cutoff SD-Rule are considered good rules and vice versa. From a linguistic standpoint, a good SD rule should remain good. We observed the good rules remain above the cutoff SD rule over the past three years. The complete log of optimization processes and SD-Rule sets for the 2014, 2015 and 2016 releases are available at the SPECIALIST Lexical Tools web site (National Library of Medicine, Lexical Tools 2014, 2015 and 2016).

System performance is used to choose the best rules from all generations in those P-C families with Parent-Rules and Child-Rules coexisting in the candidate set. As we observed, the best rules from a P-C family chosen by this computational approach concurs with the conventional linguistic standpoint. For example, the 3rd generation Child-Rule, genesis$|noun|genic$|adj, has a higher system performance than its root-parent rule, ES-16-01,

Table 4: Summary of the Optimized SD-Rule Set, 2014-2016.

Release	2014	2015	2016
No. of SD Candidate Rules	107	120	132
No. of Baseline Rules	96	101	111
No. of Good Rules	73	76	82
No. of Retrieved SD-Pairs	48,579	53,885	58,391
No. of Relevant SD-Pairs	42,552	46,950	50,814
System Precision (SP)	95.30%	95.22%	95.00%
System Recall (SR)	95.01%	95.70%	95.26%
System Performance	1.9031	1.9093	1.9026
Cutoff Rule	ar$\|adj\|e$|noun	ar$\|adj\|e$|noun	$\|noun\|ist$|noun

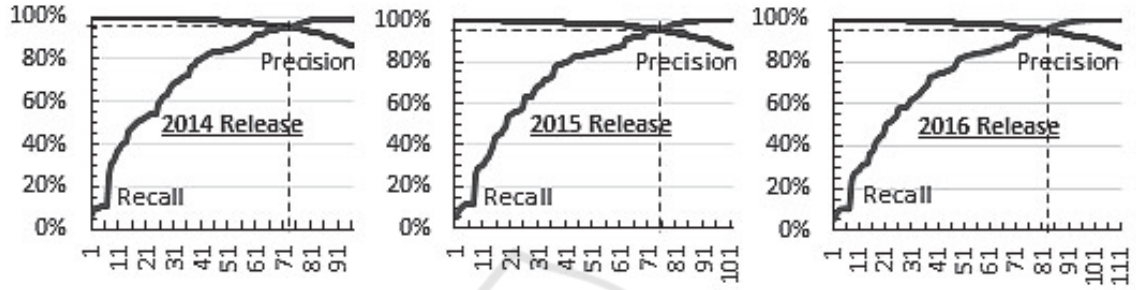

Figure 2: System Precision-Recall vs. SD Candidate Rules in the Optimized Set for Lexical Tools Releases 2014-2016.

esis$|noun|ic$|adj. This result concurs with the linguistic standpoint (Dorland's, 2003). In some cases, Parent-Rules and Child-Rules have the same system performance. For example, nce$|noun|nt$|adj and its Child-Rules (ance$|noun|ant$|adj and ence$|noun|ent$|adj) have the same system performance (highest in this P-C family). Linguistically, these two Child-Rules have the same definition and Latin suffix source. In such cases, this model chooses the Parent-Rule over the Child-Rule to keep better coverage while remaining linguistically valid

Derivational growth in the Lexicon is based on the increase of the Lexicon as well as SD candidate rules. Expert-suggested and computer-generated rules have been introduced into the system. Eight and nine new computer-generated SD candidate rules have been added to the 2015 and 2016 releases, respectively. 100% (8/8) and 78% (7/9) of them are deemed good rules in the SD-Rule set of the Lexical Tools' 2015 and 2016 releases. We plan to add more computer-generated rules for better coverage in future releases. This methodology is valuable for evaluating future SD candidate rules and is crucial when using SD-Rules in Lexical Tools.

Finding the optimized set from different SD candidate rule sets with different possible P-C rules, makes this tedious process even more complicated. For example, 11 of 13 new SD candidate rules are evaluated as good rules in the 2015 release. As shown in Table 4, the total number of candidate SD rules in 2015 is 120 (= 107 + 13). Only 5 of the 11 good rules are not parent-child related rules. Four of them have a 1-to-1 P-C relationship. For example, rule CG-15-03 in Table 1 is the parent rule of C-17 in Table 3 from 2014. These 4 rules replace old rules and do not affect the number of good rules. The other 2 good rules have 1-to-2 P-C relationships and thus reduce the number of good rules by one for each rule. For example, rule CG-15-07 in Table 1 is the parent rule of C15 and C16 in Table 3 from 2014. Thus, the total number of good rules in 2015 is 76 (= 73 + 1x5 + 0x4 – 1x2) instead of 84 (= 73 + 11). Similarly, the number of candidate rules, baseline rules and good rules in 2016 can be derived by considering P-C relationship.

This approach is a generic method to find the optimized set from a candidate set to reach the best system performance. It provides a maintainable and scalable system for generating SD-Rules in Lexical Tools to suggest derivations unknown to the Lexicon. Ultimately, it produces better precision and recall for NLP applications that use Lexical Tools derivational flow components. As the Lexical Tools' analysis of English suffix relationships thus becomes ever broader and deeper, so is its usefulness to NLP strengthened. In the future, we plan to assess the impact of our work by examining whether NLP applications show improvement using derivational features. Lexical Tools is distributed annually by

NLM via an open source license agreement.

ACKNOWLEDGEMENTS

This research was supported by the Intramural Research Program of the NIH, National Library of Medicine. The authors would like to thank Mr. Guy Divita, Mr. Howard Lu, and Dr. Fiona M. Callaghan for their valuable discussions.

REFERENCES

Aho, A.V., Ullman, J.D., and Hopcroft, J.E., 1983. Data Structure and Algorithms. Addison Wesley, pages 163-169.

Aronson, A.R. and Lang, F.M., 2010. An Overview of MetaMap: Historical Perspective and Recent Advances. JAMIA, Vol. 17, pages 229-236.

Divita, G., Zeng, Q.T., Gundlapalli, A.V., Duvall, S., Nebeker, J., and Samore, M.H., 2014. Sophia: An Expedient UMLS Concept Extraction Annotator. In proceeding of AMIA 2014 Annual Symposium, pages 467-476, Washington, DC, USA, November 15-19.

Dorland, W.A., 2003. Dorland's Illustrated Medical Dictionary, 30th edition, W. B. Saunders Company. Philadelphia, Pa, page 763.

Fung, K.W., McDonald, C., and Srinivasan, S., 2010. The UMLS-CORE Project: A Study of the Problem List Terminologies Used in Large Healthcare Institutions. JAMIA, Vol. 17, pages 675-680.

Gaussier, E., 1999. Unsupervised learning of derivational morphology from inflectional lexicons. In: Kehler A and Stolcke A, eds, ACL workshop on Unsupervised Methods in Natural Language Learning, College Park, MD. June.

Grabar, N. and Zweigenbaum, P., 2000. A General Method for Sifting Linguistic Knowledge from Structured Terminologies. In proceeding of AMIA 2000 Annual Symposium, page. 310-314, Los Angeles, CA, USA, November 4-8.

Grabar, N. and Zweigenbaum, P., 1999. Language Independent Automatic Acquisition of Morphological Knowledge from Synonym Pairs. In Proceeding of AMIA 1999 Annual Symposium, page. 77-81, Washington, DC, USA, November 6-10.

Lu, C.J., McCreedy, L., Tormey, D., and Browne, A.C., 2012. A Systematic Approach for Automatically Generating Derivational Variants in Lexical Tools Based on the SPECIALIST Lexicon. IEEE IT Professional Magazine, May/June, pages 36-42.

Lu, C.J., Tormey, D., McCreedy, L., and Browne, A.C., 2013. Implementing Comprehensive Derivational Features in Lexical Tools Using a Systematical Approach. In proceeding of AMIA 2013 Annual Symposium, Wash., DC, USA, Nov. 16-20, page 904.

McCray, A.T., Aronson, A.R., Browne, A.C., Rindflesch, T.C., Razi, A., and Srinivasan, S., 1993. UMLS Knowledge for Biomedical Language Processing. Bull. Medical Library Assoc., vol. 81, no. 2, page 184-194.

National Library of Medicine, Lexical Tools 2014, Optimizing 2014 SD-Rule Set – Add SD-Rules from Other Suggestions. Available from: <http://lsg3.nlm.nih.gov/LexSysGroup/Projects/lvg/2014/docs/designDoc/UDF/derivations/SD-Rules-Opti/ex-add-suggest.html>. (24 October 2013).

National Library of Medicine, Lexical Tools 2015, Optimizing 2015 SD-Rule Set - Optimum Log. Available from: <http://lsg3.nlm.nih.gov/LexSysGroup/Projects/lvg/2015/docs/designDoc/UDF/derivations/SD-Rules-Opti/Ex-2015/optiLog.html>. (9 September 2014).

National Library of Medicine, Lexical Tools 2016, Optimizing 2016 SD-Rule Set - Optimum Log. Available from: <http://lsg3.nlm.nih.gov/LexSysGroup/Projects/lvg/2016/docs/designDoc/UDF/derivations/SD-Rules-Opti/Ex-2016/optiLog.html>. (17 September 2015).

Pacak, M.G., Norton, L.M., and Dunham, G.S., 1980. Morphosemantic Analysis of - ITIS Forms in Medical Language. J. Methods of Information in Medicine, vol. 19, no. 2, page 99-105.

Wolff. S., Automatic Coding of Medical Vocabulary. 1987. Medical Information Processing – Computer Management of Narrative Data. Addison Wesley, Reading Mass, page 145-162.