Gender Difference in Innovation Recognition: A Textual Analysis Approach

Zihao Li[1], Shaoyu Liu

November 2024

Replication Files

1. Structure of the Archive

- **~/code**: Stores all code files
  - **run_reg.do** generates most tables in the main text and Appendix B
  - **summary_stat.ipynb** generates tables in Appendix A
- **~/rawdata**: Stores all raw datasets
- **~/cleandata**: Stores all processed datasets
- **~/regdata**: Stores all datasets used for regression analysis
- **~/patentsberta**: Stores patent embeddings and similarity scores
- **~/centrality_results**: Stores intermediate outputs on inventor centrality
- **~/network_results**: Stores intermediate outputs on inventor network distance
- **~/reg_results**: Stores all regression tables
- **~/figures**: Stores all figures
- **~/temp**: Stores all temporary datasets

2. Main Data Sources (provided in ~/rawdata)

- **PatentsView**: https://patentsview.org/download/data-download-tables
- **Kogan et al. (2017)**: https://github.com/KPSS2017/Technological-Innovation-Resource-Allocation-and-Growth-Extended-Data
- **Kaltenberg et al. (2023)**: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/YRLSKU

[1] Contact the corresponding author, Zihao Li, for any questions (zihao.li@yale.edu).

3. Replication Procedure

Generate patent embeddings

1. Run **clean_raw_patent.py**.
    - Input: rawdata/**g_patent.tsv.zip** (PatentsView)
    - Output: patentsberta/**patent_raw.csv**.
2. Run **pred_gender_patent.py**. for ind in range(0, 291). Generate embeddings.
    - Input: patentsberta/**patent_raw.csv**.
    - Output: patentsberta/patent_embedding_jsonl/**patent_embeddings_patentsberta_{ind}.jsonl**, for ind=0-291.
3. Run **gender_postprocess_embed.py**. Convert .jsonl to .npy embeddings. Generate patent dictionaries with columns [idx, patent_id, patent_year], where idx corresponds to the row index in corresponding embedding files.
    - Input: patentsberta/patent_embedding_jsonl/**patent_embeddings_patentsberta_{ind}.jsonl** for ind=0-291.
    - Output: Embeddings and patent_id.csv files.
        i. patentsberta/**embedding_0_50.npz.npy**
        ii. patentsberta/**embedding_50_100.npz.npy**
        iii. patentsberta/**embedding_100_150.npz.npy**
        iv. patentsberta/**embedding_150_200.npz.npy**
        v. patentsberta/**embedding_200_250.npz.npy**
        vi. patentsberta/**embedding_250_291.npz.npy**
        vii. patentsberta/patent_id/**patent_id_0_50.csv**
        viii. patentsberta/patent_id/**patent_id_50_100.csv**
        ix. patentsberta/patent_id/**patent_id_100_150.csv**
        x. patentsberta/patent_id/**patent_id_150_200.csv**
        xi. patentsberta/patent_id/**patent_id_200_250.csv**
        xii. patentsberta/patent_id/**patent_id_250_291.csv**
4. Run **append_raw.py**. Append files generated from the previous step.
    - Input:

i. patentsberta/**patent_raw.csv**
ii. patentsberta/**embedding_0_50.npz.npy**
iii. patentsberta/**embedding_50_100.npz.npy**
iv. patentsberta/**embedding_100_150.npz.npy**
v. patentsberta/**embedding_150_200.npz.npy**
vi. patentsberta/**embedding_200_250.npz.npy**
vii. patentsberta/**embedding_250_291.npz.npy**
viii. patentsberta/patent_id/**patent_id_0_50.csv**
ix. patentsberta/patent_id/**patent_id_50_100.csv**
x. patentsberta/patent_id/**patent_id_100_150.csv**
xi. patentsberta/patent_id/**patent_id_150_200.csv**
xii. patentsberta/patent_id/**patent_id_200_250.csv**
xiii. patentsberta/patent_id/**patent_id_250_291.csv**

- Output:
  i. patentsberta/**patentsberta_embedding_matrix.npy**
  ii. patentsberta/**patent_dict.csv**

Calculate and merge pairwise-patent cosine similarity

1. Run **compute_similarity.py**. For each patent, generate the top-k most similar patents before it, all patents before it with cosine similarity>cutoff, and a random sample of m patents before it.
   - Input:
     i. patentsberta/**patentsberta_embedding_matrix.npy**
     ii. patentsberta/**patent_dict.csv**
   - Output: For year=1981-2015, pairwise similarity files in the following form ("i" is determined by parameter chunk_size; we recommend chunk_size=500. "top_k"=101 by default; for main analysis we use k=5; for robustness check in Table B4, we vary k=1-10. "cutoff"=0.65 by default. "m_sample"=101 by default)

        i. Top k: patentsberta/patent_sim_combined/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**.
        ii. Cutoff: patentsberta/patent_sim_combined/sim_{year}_cutoff/**sim_{year}_{i}_cutoff65.csv**.
        iii. Random sample of m patents: patentsberta/patent_sim_combined/sim_{year}_sample/**sim_{year}_{i}_sample{m_sample}.csv**.
2. Run **merge_patent_sim.py**. Merge similarity files from the previous step (Argument: top_k=5 by default).
    - Input:
        i. patentsberta/**patent_dict.csv**
        ii. patentsberta/patent_sim_combined/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**
    - Output: cleandata/**sim_score_1981_2015_top5.csv**

Generate inventor network distance

1. Run **inventor_deg_dist.py** for year=1981-2015, by running **run_network.sh**. (Navigate to the “code” folder. In the terminal, type chmod +x run_network.sh, then type ./run_network.sh)
    - Input:
        i. rawdata/**g_inventor_disambiguated.tsv** (PatentsView)
        ii. rawdata/**g_patent.tsv** (PatentsView)
        iii. cleandata/**sim_score_1981_2015_top5.csv**
    - Output: network_results/**node_dist_{year}.csv** for year 1981-2015
2. Run **inventor_deg_dist_agg.py**.
    - Input: network_results/**node_dist_{year}.csv** for year 1981-2015
    - Output: cleandata/**shortest_path_sum_1981_2015.csv**

Generate inventor network centrality

1. Run **inventor_centrality.py** for year=1981-2015, by running **run_centrality.sh**. (Navigate to the "code" folder. In the terminal, type chmod +x run_centrality.sh, then type ./run_centrality.sh)
    - Input:
        i. rawdata/**g_inventor_disambiguated.tsv** (PatentsView)
        ii. rawdata/**g_patent.tsv** (PatentsView)
    - Output: centrality_results/**centrality_{year}_degree.csv** for year 1981-2015
2. Run **inventor_centrality_agg.py**.
    - Input: centrality_results/**centrality_{year}_degree.csv** for year 1981-2015
    - Output: cleandata/**centrality_sum_1981_2015.csv**

Clean-up patent data and inventor/attorney/examiner data

1. Run first part (lines 1-415) of **clean_patview_inventor.do**.
    - Input: rawdata/**g_inventor_disambiguated.tsv** (PatentsView)
    - Output:
        i. temp/**g_inventor_clean.dta**
        ii. cleandata/**g_inventor_clean.csv**
        iii. temp/**g_inventor_clean_firstname.csv**
        iv. temp/**g_inventor_clean_nomidname.csv**
2. Run first part (lines 1-104) of **genderize_patview.py**.
    - Input: temp/**g_inventor_clean_firstname.csv**
    - Intermediate output:
        i. temp/**g_inventor_firstname_gendered_r1.csv**
        ii. temp/**g_inventor_firstname_gendered_r2.csv**
    - Output: temp/**g_inventor_firstname_gendered_r3.csv**
3. Run **racialize_patview.py**.
    - Input: temp/**g_inventor_clean_nomidname.csv**
    - Output: temp/**g_inventor_race.csv**

4. Run first part (lines 1-90) of **clean_patview_patent.do**.
    - Input:
        i. rawdata/**g_cpc_current.tsv** (PatentsView)
        ii. rawdata/**g_assignee_disambiguated.tsv** (PatentsView)
        iii. rawdata/**g_location_disambiguated.tsv** (PatentsView)
        iv. rawdata/**g_patent.tsv** (PatentsView)
    - Intermediate output:
        i. temp/**g_cpc_patentlevel.dta**
        ii. temp/**g_assignee_patentlevel.dta**
    - Output:
        i. temp/**patid_year.dta**
        ii. temp/**g_patent_clean.dta**
5. Run second part (lines 420-820) of **clean_patview_inventor.do**.
    - Input:
        i. rawdata/**g_inventor_disambiguated.tsv** (PatentsView)
        ii. rawdata/**g_persistent_inventor.tsv** (PatentsView)
        iii. rawdata/**inventor_age_score_gender.csv** (Kaltenberg et al. 2023)
        iv. temp/**g_inventor_firstname_gendered_r3.csv**
        v. temp/**g_inventor_race.csv**
        vi. temp/**g_inventor_clean.dta**
        vii. temp/**patid_year.dta**
        viii. temp/**g_assignee_patentlevel.dta**
    - Intermediate output:
        i. temp/**g_inventor_gender.dta**
        ii. temp/**g_inventor_gender_race_temp.dta**
        iii. temp/**g_inventor_gender_race.dta**
        iv. temp/**inventor_age_score_gender.dta**
    - Output:
        i. cleandata/**g_inventor_gender_race_age.dta**
        ii. cleandata/**g_inventor_gender_race_age.csv**

Clean-up examiner data.

1. Run first part (lines 1-260) of **clean_patview_examiner.do**.
    - Input: rawdata/**g_examiner_not_disambiguated.tsv** (PatentsView)
    - Output:
        i. temp/**g_examiner_clean.dta**
        ii. temp/**g_examiner_clean_firstname.csv**
        iii. cleandata/**g_examiner_clean.csv**
2. Run second part (lines 110-185) of **genderize_patview.py**.
    - Input: temp/**g_examiner_clean_firstname.csv**
    - Intermediate output:
        i. temp/**g_examiner_firstname_gendered_r1.csv**
        ii. temp/**g_examiner_firstname_gendered_r2.csv**
    - Output: temp/**g_examiner_firstname_gendered_r3.csv**
3. Run second part (lines 265-305) of **clean_patview_examiner.do**.
    - Input: temp/**g_examiner_firstname_gendered_r3.csv**
    - Output: temp/**g_examiner_gender_temp.dta**

Clean-up attorney data

1. Run first part of (lines 1-255) **clean_patview_attorney.do**.
    - Input: rawdata/**g_attorney_disambiguated.tsv** (PatentsView)
    - Output:
        i. temp/**g_attorney_clean.dta**
        ii. temp/**g_attorney_clean_firstname.csv**
2. Run third part (lines 190-265) of **genderize_patview.py**.
    - Input: temp/**g_attorney_clean_firstname.csv**
    - Intermediate output:
        i. temp/**g_attorney_firstname_gendered_r1.csv**
        ii. temp/**g_attorney_firstname_gendered_r2.csv**
    - Output: temp/**g_attorney_firstname_gendered_r3.csv**
3. Run second part (lines 260-300) of **clean_patview_attorney.do**.
    - Input: temp/**g_attorney_firstname_gendered_r3.csv**

- Output: temp/**g_attorney_gender_temp.dta**

Generate additional variables

1. Run **gen_inventor_coauthor.py**.
    - Input: cleandata/**g_inventor_gender_race_age.csv**
    - Output: temp/**inventor_coauthor.dta**
2. Run **gen_leadinventor_rank.py**.
    - Input: cleandata/**g_inventor_gender_race_age.csv**
    - Output: temp/**leadinventor_rank.dta**
3. Run second part (lines 95-150) of **clean_patview_patent.do**.
    - Input:
        i. cleandata/**g_inventor_gender_race_age.dta**
        ii. temp/**g_patent_clean.dta**
        iii. cleandata/**centrality_sum_1981_2015.csv**
        iv. temp/**inventor_coauthor.dta**
    - Intermediate output: temp/**centrality_sum_1981_2015.dta**
    - Output:
        i. cleandata/**g_patent_clean.csv**
        ii. temp/**leadinventor_info.dta**
4. Run **gen_vars_gender.py**.
    - Input:
        i. cleandata/**g_inventor_gender_race_age.csv**
        ii. cleandata/**g_patent_clean.csv**
    - Output: temp/**g_patent_clean_final_g.csv**
5. Run **gen_vars_race.py**.
    - Input:
        i. cleandata/**g_inventor_gender_race_age.csv**
        ii. temp/**g_patent_clean_final_g.csv**
    - Output: cleandata/**g_patent_clean_final.csv**
6. Run **gen_vars_gender_attorney.py**.
    - Input: temp/**g_attorney_gender_temp.dta**

- Output: cleandata/**g_attorney_clean.csv**

7. Run **gen_dictionary_firm.py**.
    - Input:
        i. cleandata/**g_patent_clean_final.csv**
        ii. rawdata/**KPSS_2020_public.csv** (Kogan et al. 2017)
    - Output: temp/**firmdict.dta**

Generate Omission Panel (inventors)

1. Run **gen_omit_panel.py**
    - Input:
        i. rawdata/**g_us_patent_citation.tsv** (PatentsView)
        ii. cleandata/**sim_score_1981_2015_top5.csv**
    - Output:
        i. temp/**actual_citation_lst.csv**
        ii. temp/**omission_panel5.csv**

Generate Omission Panel (examiners)

1. Run **gen_omit_panel_examiner.py**
    - Input:
        i. rawdata/**g_us_patent_citation.tsv** (PatentsView)
        ii. cleandata/**sim_score_1981_2015_top5.csv**
    - Output:
        i. temp/**actual_citation_lst.csv**
        ii. temp/**omission_panel5_examiner.csv**

Generate Omission Panel (Robustness check of k)

1. Run **merge_kpss.py**
    - Input:
        i. rawdata/**KPSS_2020_public.csv.zip** (Kogan et al. 2017)
        ii. patentsberta/**patent_dict.csv**

        iii. patentsberta/**patentsberta_embedding_matrix.npy**
    - Output:
        i. patentsberta/**patent_dict_kpss.csv**
        ii. patentsberta/**patentsberta_embedding_matrix_kpss.npy**
2. Run **compute_similarity.py**, with inputs from the step above. Store results in a different folder (e.g. —save_dir = patent_sim_kpss)
    - Input:
        i. patentsberta/**patentsberta_embedding_matrix_kpss.npy**
        ii. patentsberta/**patent_dict_kpss.csv**
    - Output: For year=1981-2015, pairwise similarity files in the following form ("i" is determined by parameter chunk_size; we recommend chunk_size=500. "top_k"=5 by default. "cutoff"=0.7 by default.)
        i. patentsberta/patent_sim_kpss/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**
        ii. patentsberta/patent_sim_kpss/sim_{year}_cutoff/**sim_{year}_{i}_cutoff70.csv**
        iii. patentsberta/patent_sim_kpss/sim_{year}_sample/**sim_{year}_{i}_sample{m_sample}.csv**
3. Run **merge_patent_sim_kpss.py** (Argument top_k=5 by default)
    - Input:
        i. patentsberta/**patent_dict_kpss.csv**
        ii. patentsberta/patent_sim_kpss/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**
    - Output:
        i. cleandata/**sim_score_1981_2015_top5_kpss.csv**
4. Run **merge_patent_sim.py** (Set argument: top_k=10).
    - Input:
        i. patentsberta/**patent_dict.csv**
        ii. patentsberta/patent_sim_combined/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**
    - Output:

        i. cleandata/**sim_score_1981_2015_top10.csv**

5. Run **merge_patent_sim.py** (Set argument: top_k=50).
    - Input:
        i. patentsberta/**patent_dict.csv**
        ii. patentsberta/patent_sim_combined/sim_{year}_topk/**sim_{year}_{i}_top{top_k}.csv**
    - Output:
        i. cleandata/**sim_score_1981_2015_top50.csv**
6. Run **gen_omit_panel_robustk.py**.
    - Input:
        i. temp/**actual_citation_lst.csv**
        ii. cleandata/**sim_score_1981_2015_top5_kpss.csv**
        iii. cleandata/**sim_score_1981_2015_top10.csv**
        iv. cleandata/**sim_score_1981_2015_top50.csv**
    - Output:
        i. temp/**omission_panel{k}_robust.csv** for k=1-10
        ii. temp/**omission_panel_flex.csv**
        iii. temp/**omission_panel_restrict.csv**

Generate Omission Panel (Random sample of 5 previous patents)

1. Run **merge_patent_sim_sample.py**.
    - Input:
        i. patentsberta/**patent_dict.csv**
        ii. patentsberta/patent_sim_combined/sim_{year}_sample/**sim_{year}_{i}_sample{m_sample}.csv**
    - Output:
        i. cleandata/**sim_score_1981_2015_sample.csv**
2. Run **gen_omit_panel_randsample.py**.
    - Input:
        i. rawdata/**g_us_patent_citation.tsv** (PatentsView)
        ii. cleandata/**sim_score_1981_2015_sample.csv**

iii. cleandata/**sim_score_1981_2015_top5.csv**

- Output:
  i. temp/**omission_panel5_randsample.csv**

Generate Omission Panel (Cutoff based on underlying cosine similarity)

1. Run **merge_patent_sim_cutoff.py**.
   - Input:
     i. patentsberta/**patent_dict.csv**
     ii. patentsberta/patent_sim_combined/sim_{year}_cutoff/**sim_{year}_{i}_cutoff.csv**
   - Output:
     i. cleandata/**sim_score_1981_2015_cutoff.csv**
2. Run **gen_omit_panel_cutoff.py**.
   - Input:
     i. rawdata/**g_us_patent_citation.tsv** (PatentsView)
     ii. cleandata/**sim_score_1981_2015_cutoff.csv**
   - Output:
     i. temp/**omission_panel_cutoff.csv**

Generate Omission Panel (No self-citation)

1. Run **exclude_self_citation.py**.
   - Input:
     i. rawdata/**g_inventor_disambiguated.tsv** (PatentsView)
     ii. cleandata/**sim_score_1981_2015_top5.csv**
   - Output:
     i. cleandata/**sim_score_1981_2015_top5_noselfcite.csv**
2. Run **gen_omit_panel_noselfcite.py**.
   - Input:
     i. temp/**actual_citation_lst.csv**
     ii. cleandata/**sim_score_1981_2015_top5_noselfcite.csv**
   - Output:

i. temp/**omission_panel5_noselfcite.csv**

Generate Regression Panel

1. Run **gen_reg.do**.
    - Input:
        i. rawdata/**KPSS_2020_public.csv** (Kogan et al. 2017)
        ii. rawdata/**Younge-Kuhn_Patent_Families_2017-09-25** (Younge & Kuhn 2016)
        iii. cleandata/**shortest_path_sum_1980_2015.csv**
        iv. cleandata/**g_patent_clean_final.csv**
        v. cleandata/**g_attorney_clean.csv**
        vi. cleandata/**g_inventor_gender_race_age.dta**
        vii. temp/**actual_citation_lst.csv**
        viii. temp/**leadinventor_rank.dta**
        ix. temp/**leadinventor_info.dta**
        x. temp/**firmdict.dta**
        xi. temp/**omission_panel5.csv**
        xii. temp/**omission_panel5_examiner.csv**
        xiii. temp/**g_examiner_gender_temp.dta**
    - Intermediate output:
        i. cleandata/**g_attorney_clean.dta**
        ii. temp/**actual_citation_lst.dta**
        iii. temp/**KPSS_2020_public.dta**
        iv. temp/**patent.dta**
        v. temp/**patent_i.dta**
        vi. temp/**patent_j.dta**
        vii. temp/**omission_panel5_inventor.dta**
        viii. temp/**omission_panel5_examiner.dta**
    - Output:
        i. temp/**patent_family_i.dta**
        ii. temp/**patent_family_j.dta**

iii. temp/**g_inventor_gender_race_age_short.dta**
iv. regdata/**reg_panel.csv**

Generate Regression Panel (Robustness check of k)

1. Run **gen_reg_robustk.do**.
    - Input:
        i. temp/**omission_panel{k}_robust.csv** for k=1-10
        ii. temp/**omission_panel_flex.csv**
        iii. temp/**omission_panel_restrict.csv**
        iv. temp/**patent_i.dta**
        v. temp/**patent_j.dta**
        vi. temp/**g_examiner_gender_temp.dta**
    - Output:
        i. regdata/**reg_panel`k'_robust.csv** for k=1-10
        ii. regdata/**reg_panel_restrict.csv**
        iii. regdata/**reg_panel_flex.csv**

Generate Omission History

1. Run **gen_inventor_cum_omission.py**.
    - Input:
        i. regdata/**reg_panel.csv**
        ii. cleandata/**g_inventor_gender_race_age.csv**
    - Output: cleandata/**inventor_cum_omission.csv**

4. Generate Tables and Figures

Run Regressions

The tables/figures below are generated by **run_reg.do**. Input: regdata/**reg_panel.csv**. For Table 10 and B16, we also need temp/**g_inventor_gender_race_age_short.dta**, cleandata/**inventor_cum_omission.csv**, temp/**leadinventor_info.dta**.

- **Table 1: Benchmark regression**
- **Table 2: Who under-cites whom**

- **Table 3: Assignee connection**
- **Table 4: Gender-obvious vs androgynous name**
- **Table 5: Omission and future patenting**
- **Table 6: Citations by patent examiners**
- **Table B1: Heterogeneity across time period**
- **Table B2: Heterogeneity across firm tier**
- **Table B3: Heterogeneity across firm tier (alternative firm tier definition)**
- **Table B4: Heterogeneity across patent class**
- **Table B5: Heterogeneity across patent class (by cited patents)**
- **Table B6: Race and ethnicity differences in patent citations**
- **Table B7: Differences in patent citations across assignee country**
- **Table B8: Benchmark regression (showing all control variables)**
- **Table B9: Who under-cites whom (showing all control variables)**
- **Table B10: Benchmark regression with additional control variables**
- **Table B11: Validation of cosine similarity (top five similar patents)**
- **Table B17: Control for gender of the patent attorney**
- **Table B18: Cluster standard errors by cited patents**
- **Table B19: Exist_female instead of lead_female**
- **Table B20: Benchmark regression (rare-event logit)**
- **Table B21: Omission and future patenting (log(1+x))**
- **Figure 1: Inventor network**
- **Figure B1: Inventor network (no mixed-gender inventor teams)**
- **Figure B2: Inventor network (distance between closest inventors)**

The table below is generated by **run_reg_randsample.do**

Input: temp/**omission_panel5_randsample.csv,** temp/**patent_i.dta,** temp/**patent_j.dta**

- **Table B12: Validation of cosine similarity (five random patents)**

The table below is generated by **run_reg_cutoff.do**

Input: temp/**omission_panel5_cutoff.csv,** temp/**patent_i.dta,** temp/**patent_j.dta**

- **Table B13: Impose cutoff directly based on cosine similarity**

The tables below are generated by **run_reg_robustk.do**

Input: regdata/**reg_panel{k}_robust.csv** for k=1-10. regdata/**reg_panel_flex.csv**, regdata/**reg_panel_restrict.csv**.

- **Table B14: Construction of should-cite list with different values of fixed k**
- **Table B15: Construction of should-cite list with flexible k**
- **Table B16: Construction of should-cite list using KPSS firms only**

The table below is generated by **run_reg_noselfcite.do**

Input: temp/**omission_panel5_noselfcite.csv,** temp/**patent_i.dta,** temp/**patent_j.dta**

- **Table B22: Exclude self citations**

The table below is generated by **gender_doubleml/src/doubleml_gender.py**

Input: regdata/**reg_panel.csv**. See instruction: gender_doubleml/README_DML.md

- **Table B23: Double/Debiased Machine Learning**

Summary Statistics (Appendix A)

The tables/figures below are generated by **summar_stat.ipynb**

- **Figure A1: Distribution of cosine similarity by gender composition**
- **Figure A2: Distribution of KPSS index by gender composition**
- **Figure A3: Visualization of the omission index**
- **Figure A4: Female inventor representation by CPC subclass**
- **Table A1: Summary statistics of gender composition of cited patents**
- **Table A2: Summary statistics of racial/ethnic composition of cited patents**
- **Table A3: Summary statistics of relevant variables**
- **Table A4: Summary statistics of KPSS index by gender composition**
- **Table A5: Cosine similarity by gender composition of cited patents**
- **Table A6: Gender share of cosine similarity by cited patents' rank**
- **Table A7: Levels of collaboration rates within- and across-gender**
- **Table A8: Levels of collaboration rates within- and across-race/ethnicity**
- **Table A9: Total patent grants of top 30 firms**